SYSTEMS ENGINEERING
Research Center

Graphical CONOPS prototype to demonstrate emerging methods, processes, and tools at ARDEC

A013 - Final Technical Report SERC-2013-TR-031-2

July 17, 2013

Principal Investigator: Dr. Robert Cloutier, Stevens Institute of Technology

Team Members:

Dr. Drew Hamilton, Senior Researcher, Auburn University

Dr. Teresa Zigh, Senior Researcher, Stevens Institute of Technology

Dr. Peter Korfiatis, Research Assistant, Stevens Institute of Technology

Behnam Esfahbod, Research Assistant, Stevens Institute of Technology

Peizhu Zhang, Research Assistant, Stevens Institute of Technology

Patrick Pape, Research Assistant, Auburn University

Jim O'Brian, Auburn University

Sarah Weeks, Auburn University

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **17 JUL 2013**	2. REPORT TYPE	3. DATES COVERED **00-00-2013 to 00-00-2013**
4. TITLE AND SUBTITLE **Graphical CONOPS prototype to demonstrate emerging methods, processes, and tools at ARDEC**		5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Stevens Institute of Technology,Systems Engineering Research Center,Castle Point on the Hudson,Hoboken,NJ,07030**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S) 11. SPONSOR/MONITOR'S REPORT NUMBER(S)
12. DISTRIBUTION/AVAILABILITY STATEMENT **Approved for public release; distribution unlimited**		
13. SUPPLEMENTARY NOTES		

14. ABSTRACT					
This research is a continuation of research initiated in August 2011. The goal of the research was to continue the investigation of graphical 3D gaming environments in the construction of a shared mental model during concept development. A result of the research is an artifact that is a ?proof of concept? prototype, the CONOPS Navigator. The Navigator is intended to provide a 3D virtual guide through the development of a CONOPS, and also to integrate various tools and applications currently in use. This integration is a widely-sought capability, one which will enable current CONOPS developers and users the flexibility to import and export analysis parameters and results to and from various familiar and well-used tools. Legacy systems are a fact of life in operational concerns; this prototype is intended to demonstrate interconnectivity on a limited scale between specific simulation and mathematical modeling software packages, via a main operational environment. This environment was built using a game development environment. This task was always envisioned as part of a larger CONOPS research agenda. As research progressed, the potential synergies from merging RT 31 with RT 23, and then combining development architecture and strategies with RT 30, became apparent. Some already-developed external interfaces were seen as adjuncts to the activities performed in RT 30 and these interfaces would certainly be useful in the future. By far, the greater synergies in the development effort were in architecture and operational issues ? such issues are transparent to the user but vital to successful delivery. Further exploration led to an integrated data-set and application. The research includes minor updates to our approaches to implementing, managing, and addressing data impedance challenges between applications including Excel, @Risk, and MATLAB, but the research herein focuses mainly on the development of a use-case scenario-building tool, one capable of interfacing with already-existing battle simulation software. The sponsor has graciously supplied a resource to work with Unity 3D team members, to become conversant in the use of the Unity 3D modeling tool and environment. An interface with Presagis was investigated in cooperation with the Army Research, Development and Engineering Command (RDECOM) customer, and the determination is that further training in Presagis operation is required in order to interface successfully with this tool. Finally, this report incorporates much of the research from the first phase of the RT30 task. This was done to provide a comprehensive report reflecting the total research effort for future readers.					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**64**	

This material is based upon work supported, in whole or in part, by the U.S. Department of Defense through the Systems Engineering Research Center (SERC) under Contract H98230-08-D-0171. SERC is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology

ABSTRACT

This research is a continuation of research initiated in August 2011. The goal of the research was to continue the investigation of graphical 3D gaming environments in the construction of a shared mental model during concept development. A result of the research is an artifact that is a "proof of concept" prototype, the CONOPS Navigator. The Navigator is intended to provide a 3D virtual guide through the development of a CONOPS, and also to integrate various tools and applications currently in use. This integration is a widely-sought capability, one which will enable current CONOPS developers and users the flexibility to import and export analysis parameters and results to and from various familiar and well-used tools. Legacy systems are a fact of life in operational concerns; this prototype is intended to demonstrate interconnectivity on a limited scale between specific simulation and mathematical modeling software packages, via a main operational environment. This environment was built using a game development environment.

This task was always envisioned as part of a larger CONOPS research agenda. As research progressed, the potential synergies from merging RT 31 with RT 23, and then combining development architecture and strategies with RT 30, became apparent. Some already-developed external interfaces were seen as adjuncts to the activities performed in RT 30 and these interfaces would certainly be useful in the future. By far, the greater synergies in the development effort were in architecture and operational issues – such issues are transparent to the user but vital to successful delivery. Further exploration led to an integrated data-set and application.

The research includes minor updates to our approaches to implementing, managing, and addressing data impedance challenges between applications including Excel, @Risk, and MATLAB, but the research herein focuses mainly on the development of a use-case scenario-building tool, one capable of interfacing with already-existing battle simulation software. The sponsor has graciously supplied a resource to work with Unity 3D team members, to become conversant in the use of the Unity 3D modeling tool and environment. An interface with Presagis was investigated in cooperation with the Army Research, Development and Engineering Command (RDECOM) customer, and the determination is that further training in Presagis operation is required in order to interface successfully with this tool.

Finally, this report incorporates much of the research from the first phase of the RT30 task. This was done to provide a comprehensive report reflecting the total research effort for future readers.

This page intentionally left blank

TABLE OF CONTENTS

FIGURES AND TABLES

1 SUMMARY

The Department of Defense (DoD) is vigorously pursuing greater efficiency and productivity in defense spending so that it can continue to provide the armed forces with superior capabilities in an environment of flat defense budgets. Toward that end, the Office of the Secretary of Defense (OSD) has issued new acquisition guidance that places increased emphasis on system engineering early in the lifecycle to balance operational performance with affordability and has established the System Engineering Research Center (SERC) to create the tools and processes needed to execute this guidance. As one of its research areas, the SERC has put forth the notion of a concept engineering system for agile CONOPS Development.

Technical Reports SERC-2009-TR-003 and SERC 2010-TR-007 provided a compelling vision, a feasibility assessment, and an initial process definition for Graphical CONOPS development environment for agile systems engineering. Technical Report SERC-2011-TR-030 detailed the successful integration of several analysis software packages, resulting in an initial prototype which demonstrated a cohesive and easy to use collaborative concept engineering system applicable within the DoD acquisition domain.

Consistent with RDECOM's vision and mission to be the Army's primary source for integrated research, development and engineering capabilities to empower, unburden, and protect the Warfighter, this extended research topic called for the creation of another prototype – one which can adequately and easily enable stakeholders to set up and analyze actual operations, using an RDECOM-ARDEC generated scenario as a basis. The prototype was developing via the agile CONOPS development process. We expect that the prototype demonstration will guide improvements for future prototypes.

This research extends the proof of concept prototype originally dubbed the "CONOPS Navigator". This prototype provides a 3D virtual guide intended to assist one assigned to CONOPS development, through the setup of a combat scenario and the use of the Integrated CONOPS Environment Framework (ICEF). Where previous research tasks had investigated data modeling tools and the seamless transfer and manipulation of data from one application to another using Excel, @Risk, and MATLAB, the thrust of this research would be to setup and run a combat scenario. It should be mentioned that Presagis was substantially evaluated as part of this task, but not implemented due to the complexity of the Presagis interface.

2 INTRODUCTION

It is believed that the 3D gaming technologies available today can be used to provide a useful "front end" to the concept engineering process. Selection of the correct game development platform was critical to the implementation.

2.1 USE OF GAMING TECHNOLOGY AS CONDUIT FOR INTEROPERABILITY COMMUNICATIONS

In early 2011, under RT 3, gaming technology was investigated as the core backbone link between all the CONOPS-specific functionality – including scenario-building, simulation using various third-party vendor packages, and generating SysML/XML output from vendor offerings already in use by soldiers in the field. To determine which platform to select, a broad range of available gaming environments were examined:

Table 1 Game Development Engines

Torque 2D	Unreal DK	Vicious
Torque 3D	ID Tech (Doom 3)	Open Simulator
Quest 3D	Cry Engineer	C4
Unity	MS-XNA	Gamebryo
Unity Pro	Adobe Flash	Dark Basic
Unreal Engine	Source	Open Simulator

The survey examined qualitative evaluation of each platform on a number of criteria within several overall categories, as shown below:

Features/Capabilities

- Multiplayer
- 3D/2D representations
- Specific comparative strengths and limitations
- Development languages and physics engines supported

Deployment

- Client-Server capability
- Web, PC, Mac supportable
- Minimum CPU and RAM required
- Video card
- Minimum bandwidth

Compatibility with Open Source

- Source code
- Open source components
- Open interfaces

Cost:
- per seat
- to deploy
- license specifications

The evaluations of the software packages/environments along these dimensions are shown in Figure 1.

Of all the criteria evaluated, several dominated the decision-making process; most of these concerned development and deployment. These included (in no particular order):

- an active and responsive user community,
- the ability to port to different platforms easily,
- the ability to easily support multiple developers,
- providing code control (though this is not a production environment),
- supporting a diversity of programming languages transparently, and
- the ability to either have or incorporate open source components.

In today's environment of flat defense budgets, cost is also a factor, although site-wide and server licenses may help mitigate concerns that per seat licenses may incur.

Although not stated as one of the "critical" components of the decision-making process, the availability of scalable 3D models was also crucial. The applications will be operating in (and as) a visually-based immersive environment; having the models and simulation as realistic as possible will help increase the probability of acceptance and usage by the eventual field users. 3D models can also have a considerable cost factor. For the initial RT-31 task, the group utilized 3D models that were found at no cost. For RT-31a, several models needed to be purchased, to represent actual soldiers carrying relatively realistic-looking weaponry. Although the selected platform does have extensive libraries of 3D models, most are available at a nominal fee. Those models requiring animation almost always cost more money.

Most of the platforms also had other limitations, another factor when selecting the platform – cost and point-of-view being two major considerations.

NAME	FEATURES/CAPABILITIES					DEPLOYMENT									OPEN SOURCE FRIENDLY			COST		
Tech	multiplayer	3D/2D	Strengths	Limitations	development features	Client-Server	Web	PC	Mac	OS	min. CPU	min. RAM	video card	min net BW	source code	open source components	Open Interfaces	cost per seat	Cost for deployment	already own
Torque 3D	yes	3D		support, no dev exp	Physics, art pipeline		yes	yes	yes						yes (extra)	Supports all formats, Blender, Maya,		$1000-more for externally funded?		yes
Torque 2D	yes	2D			Physics, C++ like scripting		yes	yes	yes						yes (extra)			$1,250		yes (source)
Quest 3D	yes	3D					yes	yes	no						?			$3,150		yes
Unity	yes	3D & 2D			Plugin, standalone, physics engine, texture, shading, javascript, C#, Boo (Python)		yes	no	no						no	AV Codec (ogg), .Net, Server add-o		Free To < $100k		yes
Unity Pro	yes	3D & 2D	experience, community		Plugin, standalone, physics engine, texture, shading, javascript, C#, Boo (Python)		yes	yes	yes						yes (extra)	AV Codec (ogg), .Net, Server add-o		$1,200		yes
Unreal Engine	yes	3D & 2D		**1st person shooter, cos**	Physics engine, lighting, shadows, C++		no	yes	no						yes			> $700k		no
Unreal DK	yes	3D & 2D		**1st person shooter**	Physics engine, lighting, shadows, C++		no	yes	no						no			$2500/seat/year		no
CryEngine	?	3D		**cost, hardware requirements**			no	yes	no						yes			Free (educational / research)		no
MS XNA	yes	3D & 2D	free	**public, users must pay**	.Net, DirectX, asset pipeline, rendering		no	yes	no						no			free, membership required to play		no
Adobe Flash	yes	2D	ubiquitous plugin	2D, not physics based			yes	yes	yes						no, but projects can be			$449		yes
Source	yes	3D		**cost, 1st person shoote**	Direct3D / OpenGL rendering, facial animation, skeletel animation, physics engine		no	yes	yes						yes			?		no
Vicious	yes	3D & 2D			libraries, strong support for AI		no	yes	no						yes			?		no
idTech (Doom3)	yes	3D		**cost, 1st person shoote**	bump and normal mapping, skeletal animation		no	yes	yes						no (possibly in 2011)			?		no
Gamebryo	yes	3D	RPG	**likely cost**	rapid prototyping, extensible infrastructure, scripting tools		no	yes	no						yes (extra)			?		no
Dark Basic	yes	3D & 2D		limited community, assets	simple scripting of DirectX, camera, lighting, library of commands		no	yes	no						no			$40		yes
Open Simulator	yes	3D	open source	not a full engine, **alp**	physics simulation, multiple engines, multiple clients and protocols		yes	yes	yes						yes			free		no

Figure 1 Evaluation of Serious Gaming Technologies

2.2 Final Platform Selection

As can be seen in Figure 1, many of the investigated platforms have major drawbacks (shown in red). Chief among these was their inability to deploy on the Web. A secondary consideration for this phase of the research task is the ability of the tool to interface with open source code and components.

The selected platform was Unity 3D Pro. Being more intuitive, the learning curve for developers was found to be less daunting than that of most of the other platforms, and the facility to develop and deploy components was relatively easily-acquired.

Unity 3D Pro has an asset server which acts as a central code storage and a rudimentary code control mechanism. It has a rich library of models, environments, scripts, and other development components available, either free or at a nominal cost.

Unity 3D Pro supports a number of programming languages: C#, Boo (Python), and Javascript. The Unity physics engine supports movement, collision and gravity for solid objects, and users can modify textures/meshes. This ability will be critical if terrain generation from various USGS databases is to be evaluated.

Unity 3D Pro has a large user community which is extremely responsive to posted questions, and a forum containing posted solutions to many commonly-found problems or desired effects. As this research task was focused mainly on interfaces between 3rd-party software, the research team did not find solutions in user community resources for these tasks, however the resources did help when implementing some of the more complex model representations and movement.

Finally, using Google Trends (http://www.google.com/trends/) , which tracks the trend of search queries, showed a steady upward trend for those searching for information on Unity 3D as seen in Figure 2.

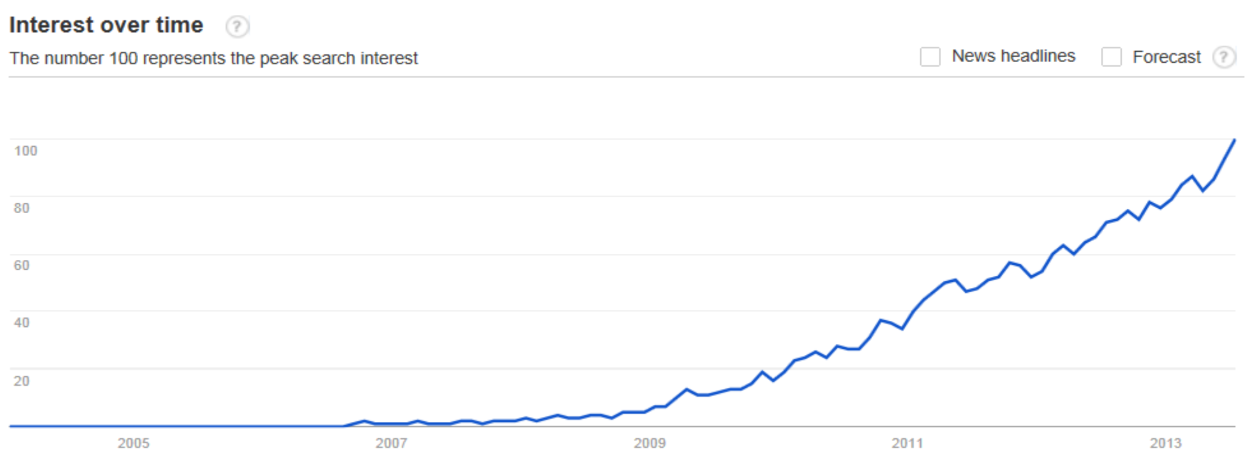

Figure 2 Google Trends for "Unity 3D"

3 Work Performed

Table 2 RT31 Sponsor Meeting Schedule

Kickoff	08/13/2012
Interim Discussions	Ad-hoc, as needed
Weekly Discussions with programming staff	From 10/2012 – 3/2013
Final Review/Presentation	07/17/2013

This work was performed in one stage, although the tasks were partitioned by team location and relative strengths:

- The Auburn University team performed extensive research into the implementation of animation in the application.
- The Stevens team worked to leverage the benefits gained from RT-30 and RT-30a to incorporate scene-building and use case modeling into this research task.

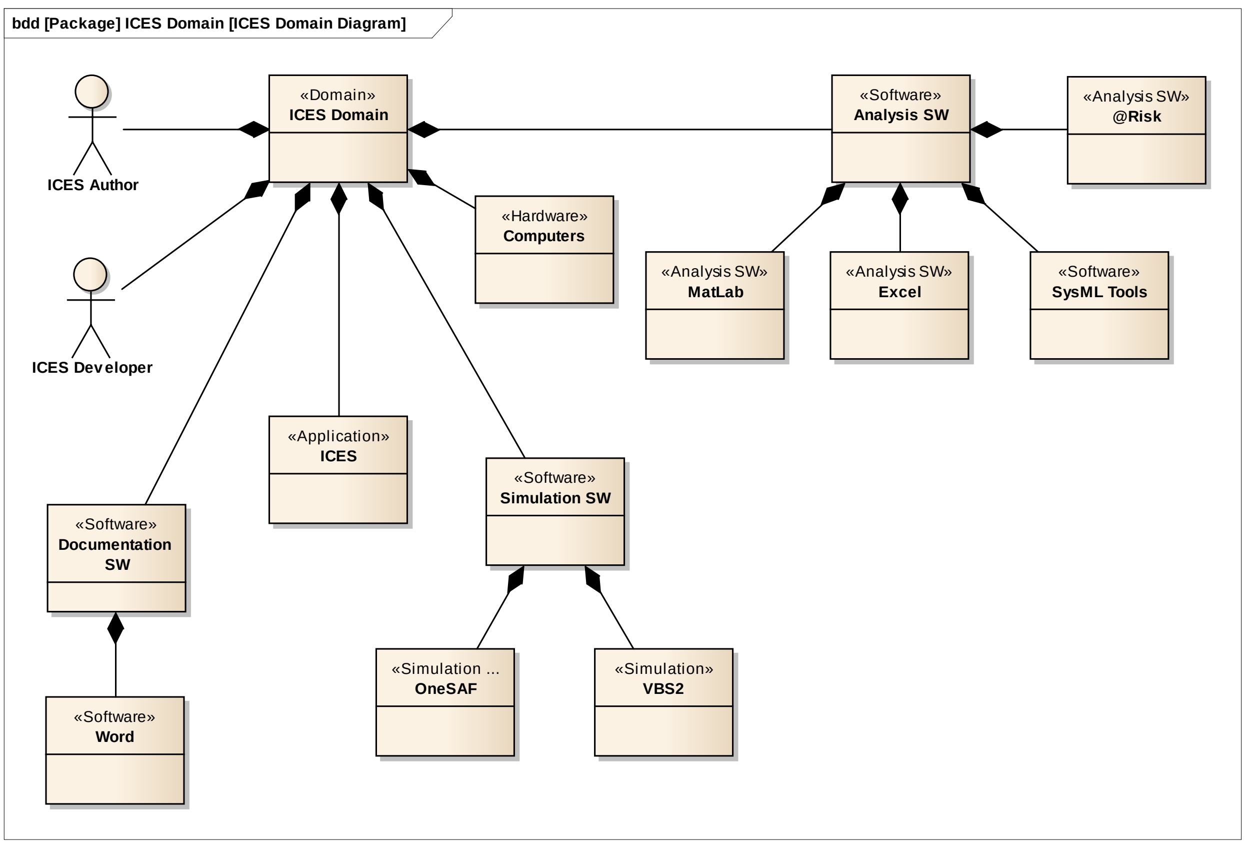

Figure 3 Original CONOPS Navigator Domain

Figure 3 and Figure 4 show the external interfaces for the CONOPS Navigator at the end of phase 1.

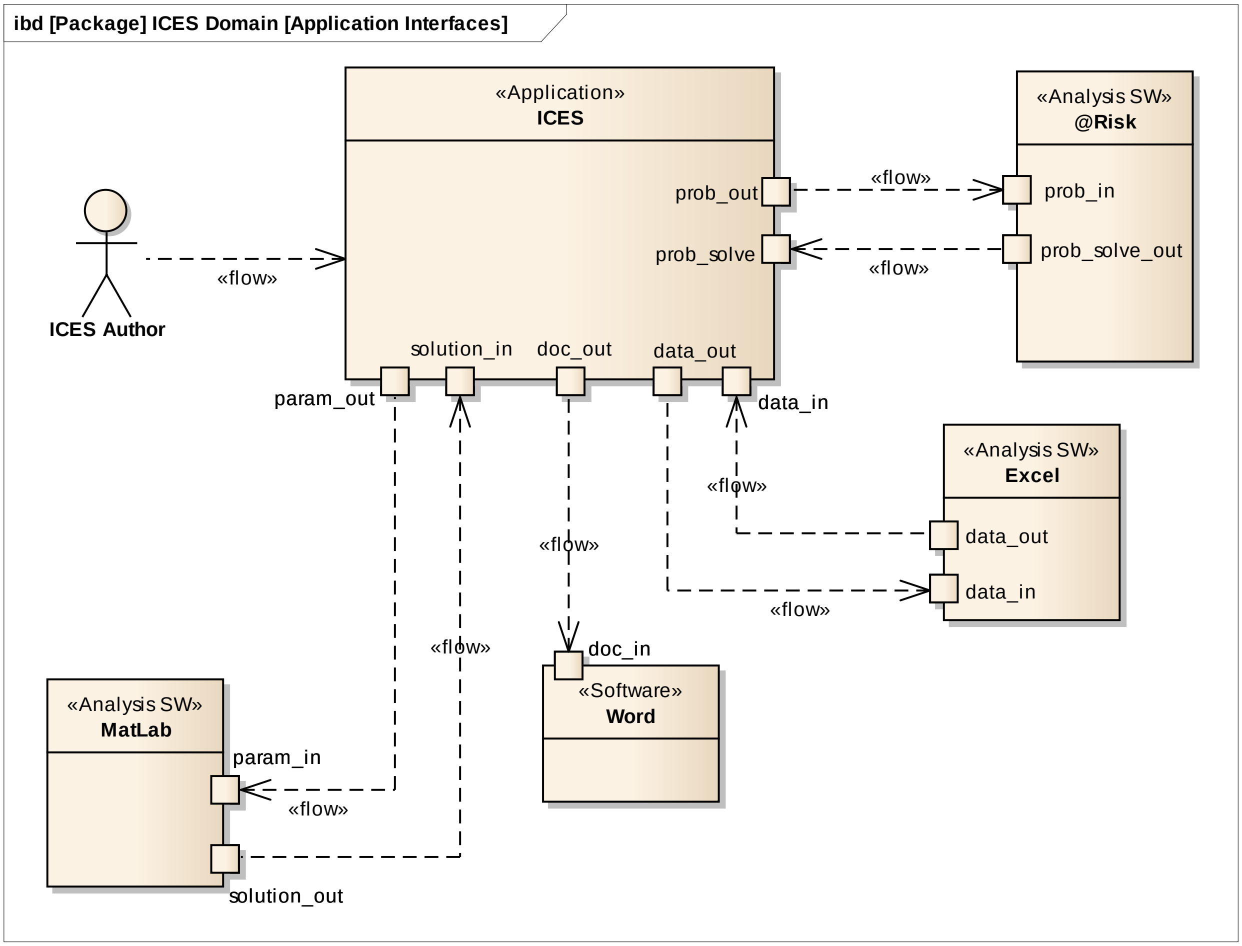

Figure 4 Original CONOPS Navigator External Interfaces Diagram

This is a proof-of-concept research task – that implies that the software must perform within a relatively flexible set of criteria; it is not a production system. Of necessity, major error-handling is not a factor in the evaluation of preparedness, but reasonable error-handling and performance issues are addressed. It should be mentioned that the current architecture and developed executable programs have proven to be quite robust – repeated testing with end-users has resulted in no system disruptions or crashes.

Although we could capitalize on the interface between Unity and Excel to perform our use-case simulations, it was felt that this layer of complexity would reduce performance time. All simulation calculations are therefore performed within Unity, and the output of RT-31 is still accessible in raw files.

The CONOPS Lobby is a virtual room where a user could choose among several options for their particular need (i.e., Microsoft Excel, @Risk Simulation libraries, Sparx (SysML package), MATLAB (via the Decision Support Center, DSC). The main thrust of this research was to enable the sponsor to model a squad-centric use case.

The original tool interfaces from the CONOPS lobby is shown in Figure 5.

Figure 5 CONOPS Lobby

3.1 Excel – Interface & Operation

Upon the selection of Excel, the following right- and left-hand side menus appear Figure 6:

Figure 6 Excel Input

The software allows the user to specify a data file for input. Once entered, as shown below, the user can select from various result options. Below the output resulting from the selection of all the available general statistics for the dataset provided in the test file is displayed in Figure 7:

Figure 7 Excel Output – General Statistics for test data file

The user is then given the option to export the results data directly to a file which can be stored, or to open the results data in a Microsoft Word document, for further viewing or possible manipulation (Figure 8 and Figure 9).

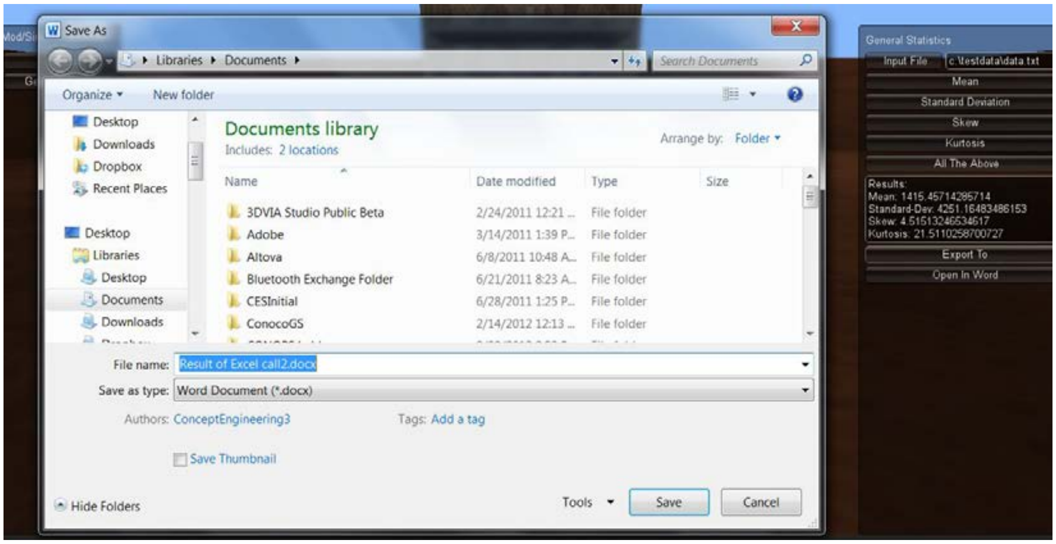

Figure 8 Browser for storing output as Microsoft Word Document

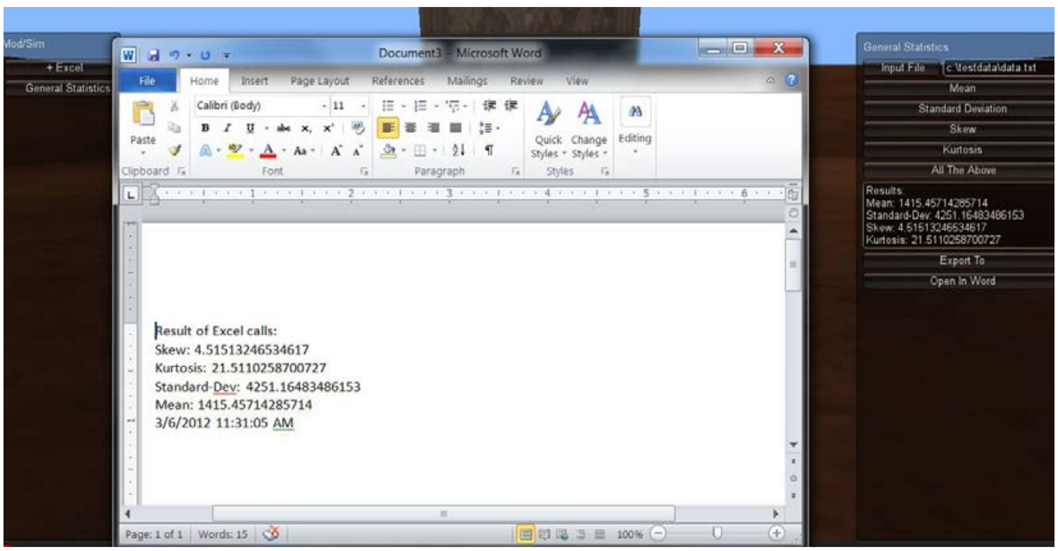

Figure 9 Exported data to Microsoft Word document

The use of Excel is enabled by C# scripts within Unity 3D Pro, and uses two external programs for initiating IO Pipes. The two external programs reside in a special Deploy folder, and must be present for the application to successfully call the Microsoft Excel functions, as well as writing to a Microsoft Word document. This is an example of the synergy of this development, as well as the benefits of using named pipes. A named pipe is an extension of the pipe concept on Unix-type systems, and serves as the inter-process communication conduit for the data stream input and output. A named pipe is system-persistent and exists beyond the life of the process, which requires that it be deleted once it in no longer needed. Once the process connects to the named pipes, communication between applications is possible.

3.2 @RISK SIMULATION

The selection of the @Risk Simulation libraries leads to similar input screens, although they are tailored for individual input - characteristics of the distributions which serve as input to the libraries.

Figure 10@Risk Simulation - Output of LogNormal Distribution

Figure 11 @Risk Simulation - Output of PERT Distribution

The calls to the @Risk simulation SDK libraries are made via Javascript. The return values are text, and the graphic representation is also formatted as a stream of text (Figure 10 and Figure 11).

3.3 Decision Support Center

The decision support application is partitioned into three sections, each of which highlights a separate interface.

3.3.1 Vehicle Simulation

Upon selection of the Decision Support Center application, Vehicle Simulation, the following initialization screen is displayed (Figure 12).

Figure 12 Vehicle Simulation Initial Screen, MATLAB initialization being performed (JLTV shown)

The user can use the slider bars shown in the above figure, to vary the distance of the simulation, the speed and acceleration of the vehicle. The application retrieves vehicle specifications and parameters from an Excel file. In this file, each sheet represents the specifications of a vehicle – the file can be extended and modified as necessary for additional vehicles (see Figure 13 below).

Vehicle 1			
Name:	Humvee		
Vehicle Attributes			
Speed Distribution	RiskLognorm		
Speed (mean)	46		
Speed (std dev)	12		
MPG	20		
Personnel Capacity	2		
Max Speed	65		
Max Acceleration			

Capacity model			
Internal Variables		*External Variables*	
Personnel Capacity	2	Required Passenge	53
MPG	20	Distance	125
		Cost of Fuel	3.3
Calculations			
Trips Required	27	Trips	
Total Distance to Travel	6750	Miles	
Fuel Use	337.5	Gallons	
Fuel Cost	1113.75	Dollard	
Travel Time (average)	146.7391304	hours	

Fuel Economy Model			
Internal Variables		*External Variables*	
MPG	20	Distance	125
		Cost of Fuel	3.3
Calculations			
Fuel Use	6.25	Gallons	
Fuel Cost	20.625	Dollars	

Response Time Simulation			
Internal Variables		*External Variables*	
Distribution Type	RiskLognorm	Distance	125
Speed mean	46	Cost of Fuel	3.3
Speed SD	12		
MPG	20		
Calculations			
Average Speed	60	MPH	
Response Time	2.083333333	Hours	
Fuel Use	6.25	Gallons	
Fuel Cost	20.625	Dollars	

Figure 13 Sample Excel Vehicle Definition File

The application also shows an initialization of MATLAB, prior to running the application. If MATLAB is not installed, the user will not be able to run the simulation. The initialized application is shown in the next three figures; the first is a 3rd-person view (Figure 14), the second is the overhead point of view built into the application (Figure 15), and the third is a 1st-person "driver" view from the vehicle interior (Figure 16).

Figure 14 Vehicle Simulation Initial Screen, MATLAB verified (MRAP shown) 3rd Party POV

Figure 15 Vehicle Simulation, Overhead POV camera

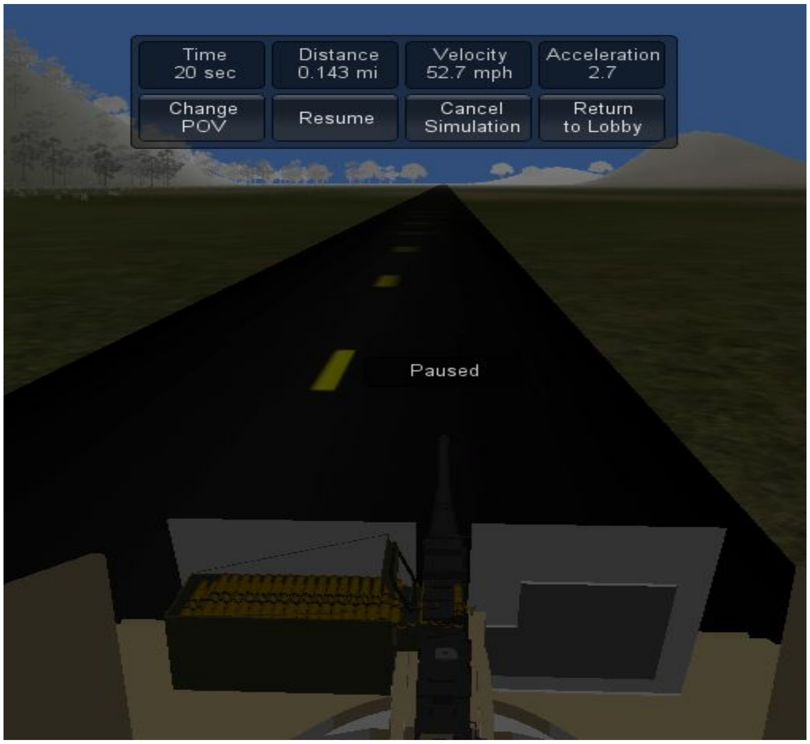

Figure 16 Vehicle Simulation Driver POV

3.3.2 Modeling the Vehicle Motion in MatLab

The algorithm to model the ideal one-dimensional motion of a vehicle over a specified distance assuming a maximum velocity, acceleration, and jerk used in this simulation is based loosely on the work of Richard D Peters (Peters). This algorithm runs iteratively calculating the parameters to model the vehicle at each step of time and accounts for the four possible outcomes of motion:

- max velocity is reached,
- max acceleration is reached but not max velocity,
- neither max velocity nor max acceleration is reached, and
- max acceleration is not reached but max velocity is reached.

The MATLAB program was then integrated with the Unity platform to show a real time representation of this data in a visual simulation. Unity creates a TCP/IP listening server, opens MATLAB, connects as a client to the Unity application on the specified port, sends a request for data, and then waits. During this time, the user on the Unity application is given time to choose a vehicle, distance, max velocity, and max acceleration. Once MATLAB initializes, the user is then given the option to run the simulation. As the simulation button is pressed, data is passed through the TCP/IP connection to MATLAB which interprets the input and begins running the simulation.

On each iteration MATLAB first checks for a command from Unity, then calculates the next set of parameters, and sends them to Unity over the network connection. As Unity gets the data packets, it converts them to the distance velocity and acceleration arguments, moves the vehicle appropriately on the next frame and updates the display for current position, velocity, and acceleration. At any point during the simulation, the user can pause the simulation, restart the simulation with the same or different parameters, or cancel the simulation and exit to the main menu. This is achieved by sending a command packet to MATLAB over the established TCP/IP connection and allowing the MATLAB program to process the command and act accordingly.

The current basic formulation of the MATLAB model does not yield overly powerful results, but it proves the concept of a real time simulation built around the computational power of MATLAB and the visual properties of Unity.

Future simulations could include more powerful formulations and one investigation can include a feedback loop from Unity. For example, a more complete model could be created for the vehicles, including properties like torque and mass. A 3D path could be created in Unity for the vehicle to follow and, as the vehicle moves along that path, data could be sent to MATLAB concerning the pitch and yaw of the vehicle, which would affect its velocity and acceleration characteristics. As this data is sent to MATLAB in each frame, the subsequent calculation would be sent back showing new displacement acceleration and velocity in each direction as well as about each axis.

3.3.3 Vehicle Allocation

Upon selection of the Decision Support Center application, Vehicle Allocation, the user can select the comparison of vehicles for various parameters, the first one shown below, is vehicle carrying capacity – in this case, between a Humvee and a JLTV (Figure 17) and (Figure 18).

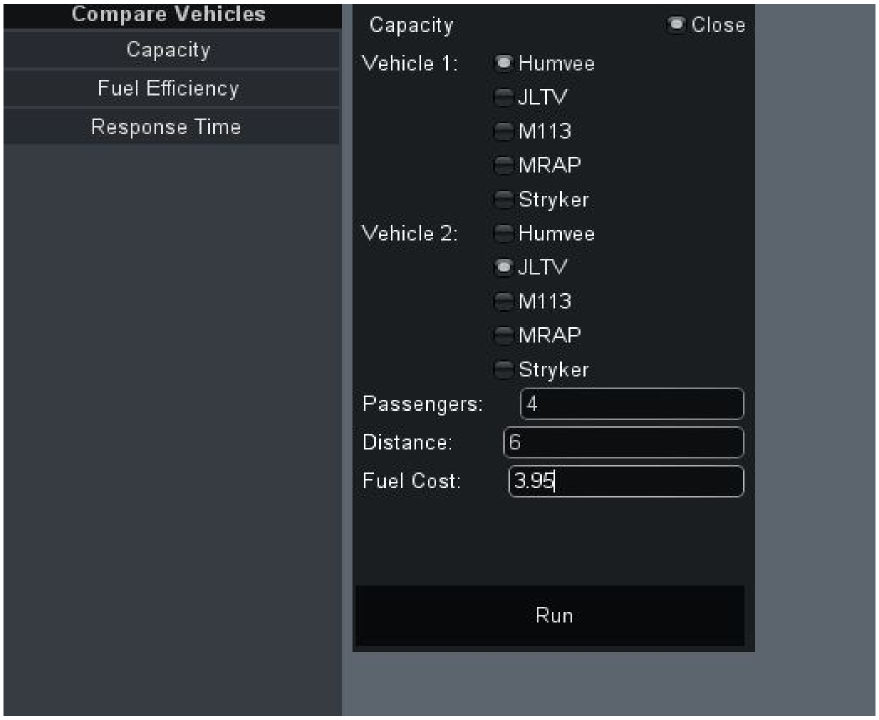

Figure 17 Vehicle Capacity Input Screen - Comparing Humvee and JLTV

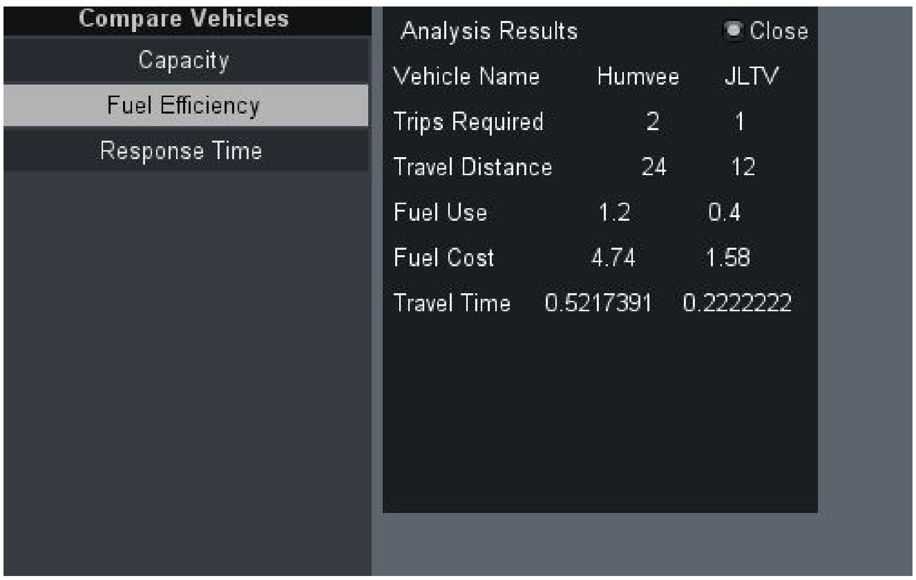

Figure 18 Vehicle Capacity Output

In order to run the vehicle fuel efficiency calculations, the initial screen presented (Figure 19) is:

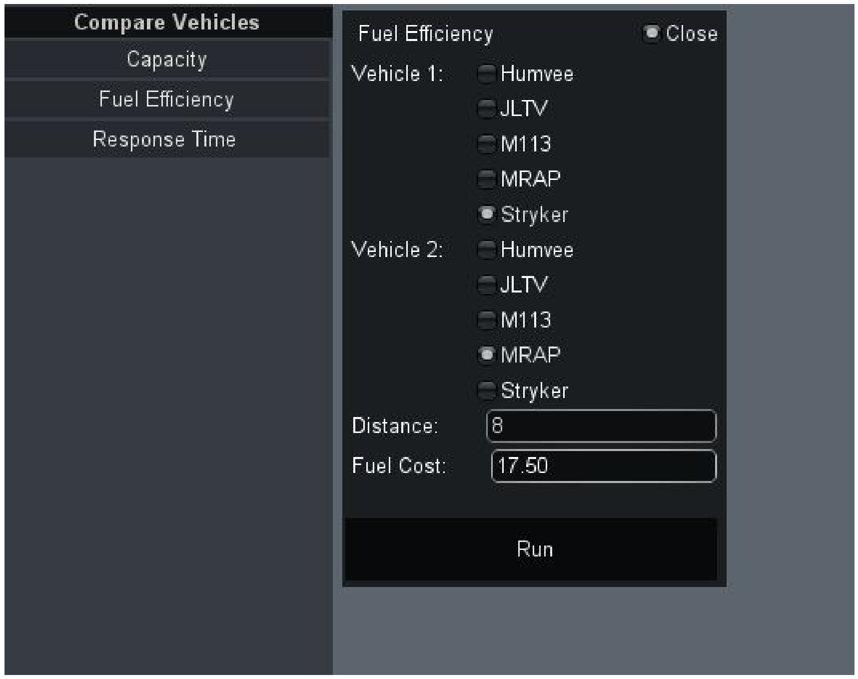

Figure 19 Vehicle Fuel Efficiency Initial Screen

In this case, the vehicles being compared are a Stryker and an MRAP, over a distance of 8 miles and with a fuel cost of $17.50/gallon. The output from this simulation is shown in Figure 20:

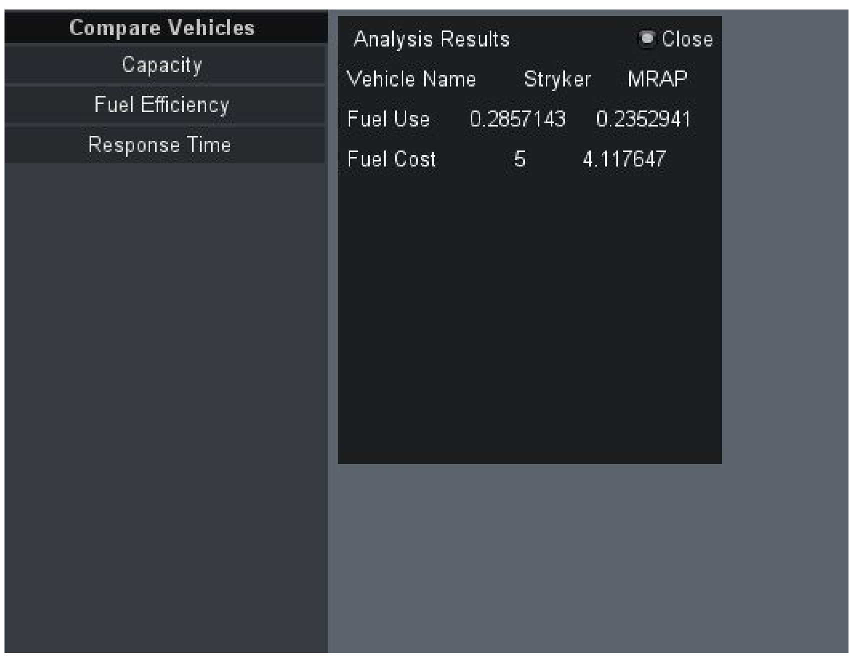

Figure 20 Vehicle Fuel Efficiency Comparison Output

3.3.4 Response Time

Upon selection of the Decision Support Center application, Response Time, the user can compare the fuel usage and fuel cost between two vehicles traveling the same distance (Figure 21 and Figure 22).

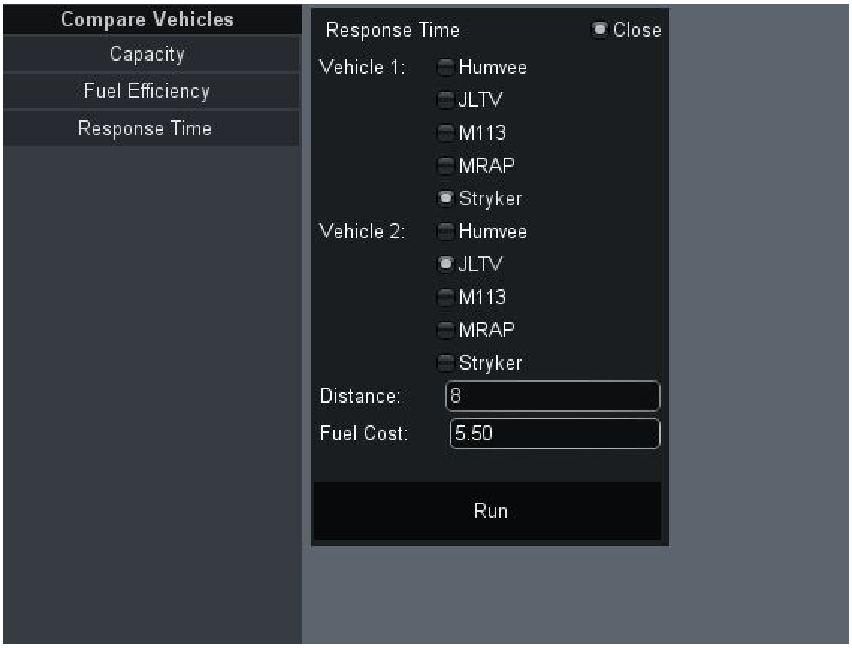

Figure 21 Response Time Input Screen

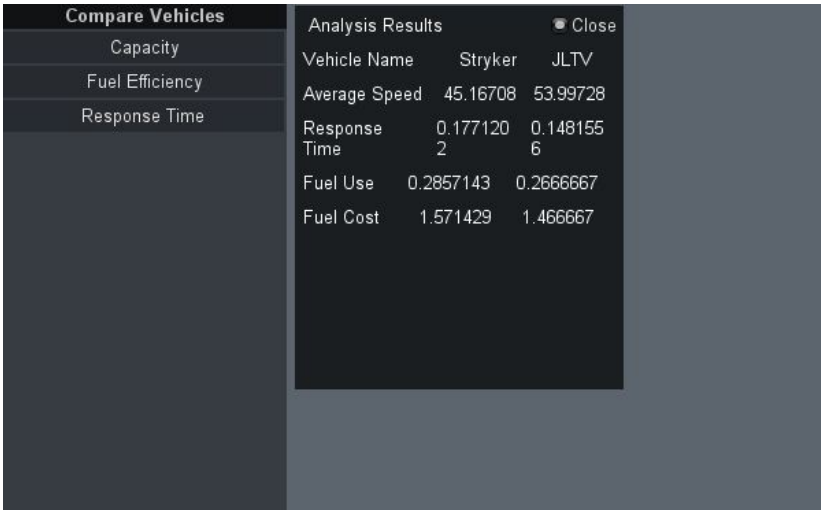

Figure 22 Response Time Output Screen

The calculations for the vehicle comparison Capacity and Fuel Efficiency decision components are being made via the Excel interface. The Response Time simulation is handled using the @Risk Simulation SDK library function, as a probability distribution is used to specify average speed.

3.4 USE CASE SCENARIO MODELING

The Integrated Concept Engineering Framework (ICEF) also provides the ability to model use cases specific to end user's needs. This capability leverages work performed under RT-30 and RT-30a, the ICEF architecture, the modeling of a new domain within the architecture, and the modifications needed to support the new domain.

3.4.1 ARCHITECTURE OF ICEF SOFTWARE

The ICEF architecture provides flexibility, reusability, and extensibility for this research tool. ICEF subsumes the original ICEF capabilities and interfaces, and implicitly generates structured data, which can then be shared and visualized among all stakeholders. Shown below is an overview of the architecture.

The ICEF requires a clear delineation between data and user interaction. This was accomplished by implementing the well know Model-View-Controller pattern, shown in Figure 23.

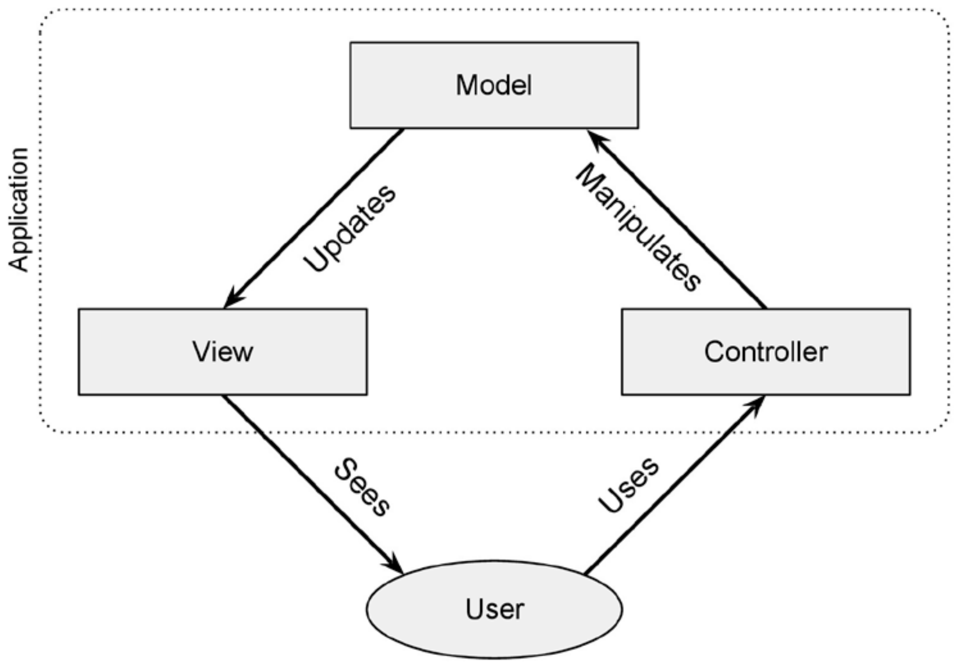

Figure 23 Basic Model-View-Controller pattern with relationship to user

The View manages the graphical output, the Controller interprets user inputs and command the model to change as appropriate, and the Model manages the behavior and data of the application, responds to requests for information about its state, and responds to instructions to change states. This separation of responsibilities is necessary to ensure scalability as well as stability in graphical user interfaces.

Because the ICEF is a real-time application with remote data sharing capabilities, the Model-View-Controller pattern is used in the client application where the line between the user interface and the pure data is drawn. There are two loops in the ICEF architecture (Figure 24 and Figure 25), since the software has both 2D and 3D interfaces.

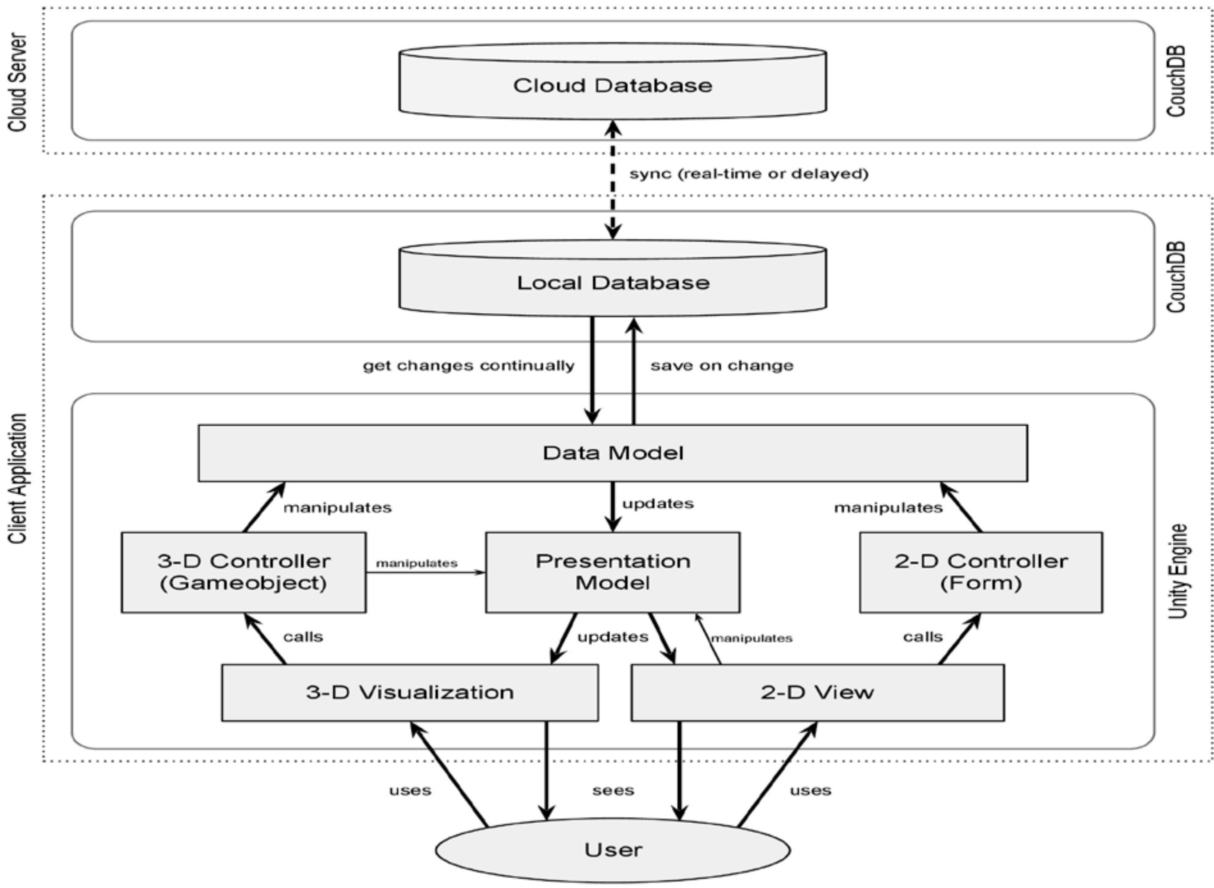

Figure 24 ICEF Architecture

The ICEF logical internal architecture is shown below.

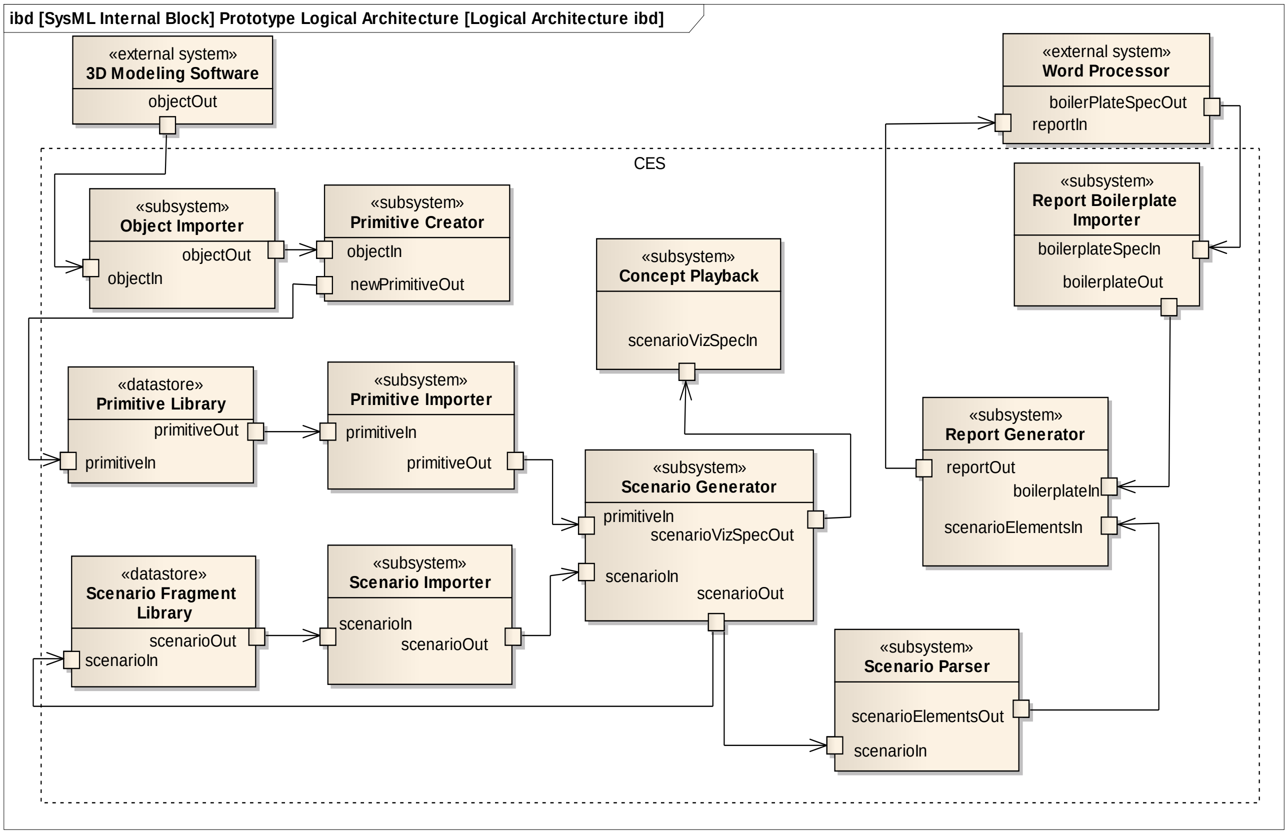

Figure 25 ICEF Internal Architecture

3.4.2 Data and Presentation Models

Within the ICEF environment, there are two different models – one to handle the domain data that is being stored and shared between users (the Data model) and one to handle the execution of a specific application (the Presentation model). The Data model has classes, shown in Table 3 below, which relate to the storyboarding concepts discussed in Section 3.4.3 Terminology of ICEF & Storyboarding Techniques.

Table 3 ICEF Data Class Model Listing

Data Model Class	***Description***
Building	The environment of a location, including its 3-D model
Domain	The domain as specified by Actors and Actions
Link	Connects actors of an action (if any)
LinkType	Indicates the type of a link (the verb)
ObjType	Class type of an actor, defining its behavior (e.g. Person, Information, Equipment)
Primitive	The specific class of an actor, defining its Domain, ObjType, and the 3-D model associated with it
PrObject	Represents the Actor, defining its Primitive and membership relationship in any Organizations or Teams
PrObjPos	The position of an Actor during a specific Action
ScAction	An Action, specific to one or more domains, may contain a Link, which defined the Actors involved in the Action
Scenario	Contains an ordered list of Scenes and may have a human-readable summary
Scene	Specifies a location, some Actors, an ordered list of Actions, and may have a human-readable description
ScnrTalk	The conversation shared between Scenario authors

During user workshops, the need to add additional Actors or Actions during CONOPS development was identified. This feature was added by creating an additional field and allowing the user to mark the actor as a placeholder for objects or actions which do not have an available 3D model (for Actors) or activity listing (for Actions).

The relationship between the data model classes is shown below in Figure 26.

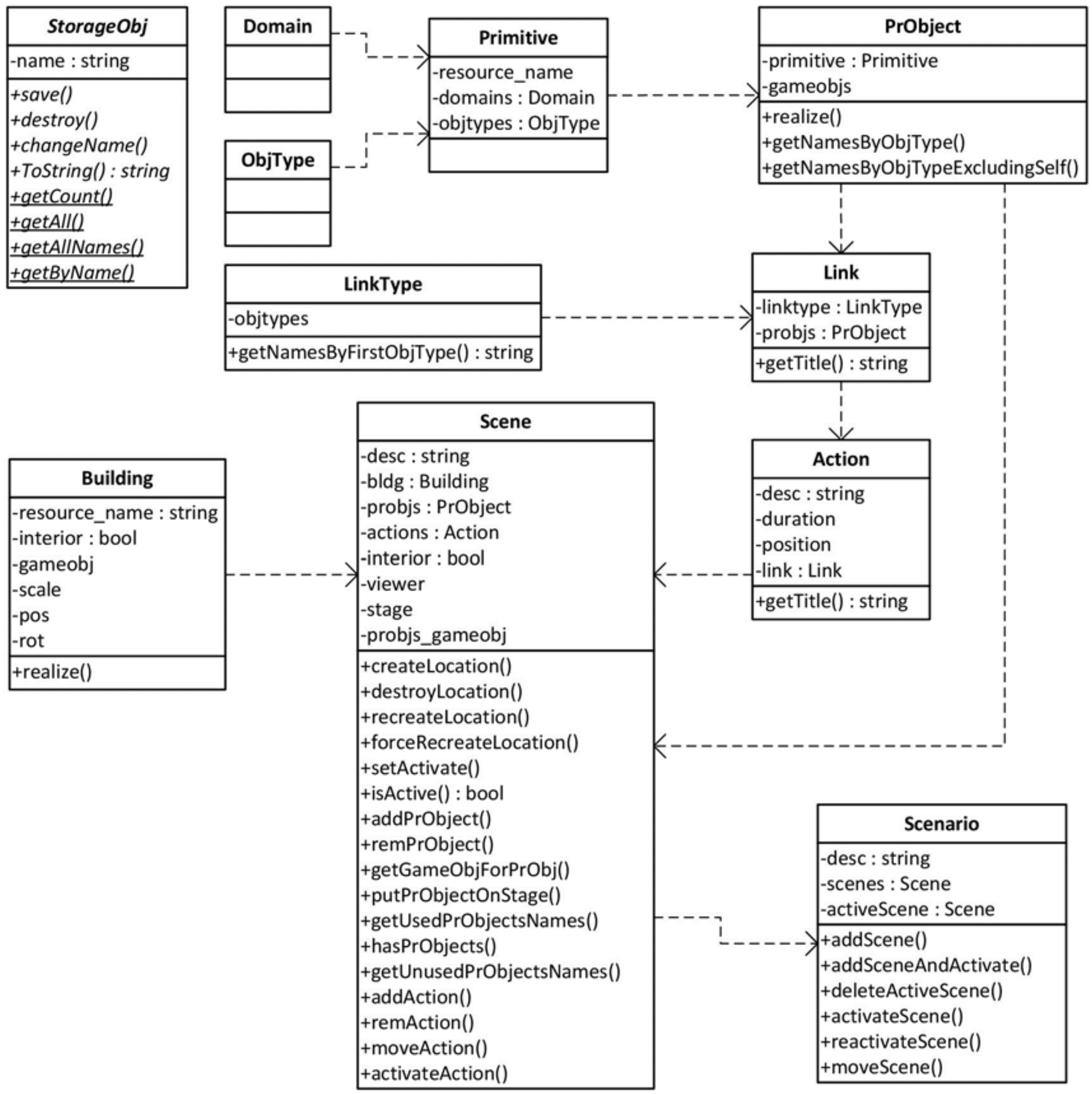

Figure 26 Data Model Class Relationships

3.4.3 TERMINOLOGY OF ICEF & STORYBOARDING TECHNIQUES

The idea of storyboarding comes naturally to the discussion of CONOPS, and the application takes the unyielding and relentless march of linear time and adapts to it. As a scenario unfolds, the viewer can see them develop in a natural way. However, when authoring a scenario, greater flexibility is needed – there may be many modeling iterations and many different configurations tested before a suitable CONOPS can be modeled and then generated. This requires that the author be able to view a graphical representation of the scenes which comprise the scenario, as well as their sequence and the order of actions within each separate scene. Some of the frequently used terms are shown below.

Scenario - a set of activities comprising the use case of a system, containing the functional flow from the system user perspective

Scene – an ordered set of actors and actions, in a specified location.

Location – the geographical site where actors congregate and actions occur. Each scene is bounded to one location, but a single location may appear in many scenes.

Actor – the basic elements of a system which are involved in actions. An actor may be a person, a device, information, etc.

Team & Organization – groupings of basic elements, and a new entity which can participate in actions. Actors have a membership relationship with teams and organizations.

Actions – the building blocks of scene flows, they describe hour activities are executed in a scene. Actions are a partially ordered set, and can be projected onto a one-dimensional list. Concurrency of actions is possible.

Duration – the length of time an action pertains. The duration of a scene is the minimum time required to complete all the contained actions.

3.4.4 Exporting CONOPS to SysML - Tool Interoperability

An important part of building any engineering tool is interoperability with other engineering tools used during the development lifecycle. In fact, one of the major challenges laid forth by the INCOSE Model-Based Systems Engineering initiative has been interoperability of model-based systems engineering tool across the system development lifecycle. Since ICEF is among the first tools of its kind, attempting to extend the principles of MBSE to the earliest parts of the system engineering lifecycle, emphasis was placed on enabling interoperability with industry standard SysML tools.

Looking at natural language equivalents of the ICEF terminology, a link can be extended to matching SysML entities and the diagrams for which they are applicable. This relationship can be seen in Table 4. This table makes use of the following abbreviations for specific SysML diagrams:

Structural Diagrams

- bdd = block definition diagram
- ibd = internal block diagram

Behavioral Diagrams

- uc = use case diagram
- act = activity diagram
- par = parametrics diagram

Table 4 Translating natural language to ICEF to SysML

Natural Language	ICEF Component	SysML Entity (diagram)
Noun	Object (system, system components)	Block (bdd, ibd)
Person	Actor (user, stakeholder, person)	Actor (bdd, uc)
Verb	Actions (user/system action & reaction)	Activity/Use Case (act/uc)
Property	Attribute (performance parameter)	Property (bdd, par)
Team	Team (groups, organizations)	Aggregation Relationship (bdd)

This mapping of ICEF components allowed for translation between ICEF scenarios and SysML entities. Additionally, given the storyboarding structure of ICEF scenarios, the user-entry field for scenario and scene were used to define high level use cases in the uc diagram.

With this mapping in place, ICEF developers added the capability to export to SysML. At any time during the ICEF modeling session, a user can click the button labeled "Export SysML for Scene" and ICEF will query the CouchDB database entries for each scene and generate an XML document. The user can then open the SysML tool Enterprise Architect, import the XML files, and SysML bdd, uc and act diagrams will be generated. A challenge in this capability is the variance of XML schema required by specific SysML tools. The XML file produced by ICEF matches the schema required by Enterprise Architect, which was chosen due to its simplicity. However, a number of other commercial SysML tools provide modules that can convert Enterprise Architect models to other, more complex XML schemas. For example, an XML file generated from an ICEF scenario was successfully imported to Enterprise Architect, and using the NoMagic Enterprise Architect converter, the resulting diagrams were imported to MagicDraw. However, some revision and rework is needed to "fix" the translated diagrams with MagicDraw, especially the act diagram.

With this capability, users who model operational scenarios within ICEF can export contents of the scenes directly to a SysML tool. Although the functionality has not been integrated into ICEF yet, this mapping of ICEF components to SysML diagrams would also allow for translation of SysML diagrams to ICEF models.

3.4.5 Scenario Building

In RT-30a, the use cases modeled separate "rooms" – each room signifying a unique and distinctive locale. This was reflected in the bottom-most portion of the author's screen, where the scene sequencing was visible. In RT-31a, however, the use case examined was a tactical contact-to-fire scenario, in which the locale remained static and the movement of the camera determined which specific section of the locale served as the "room" of a scene (Figure 27). This approach capitalizes on the already-existing scene construction model.

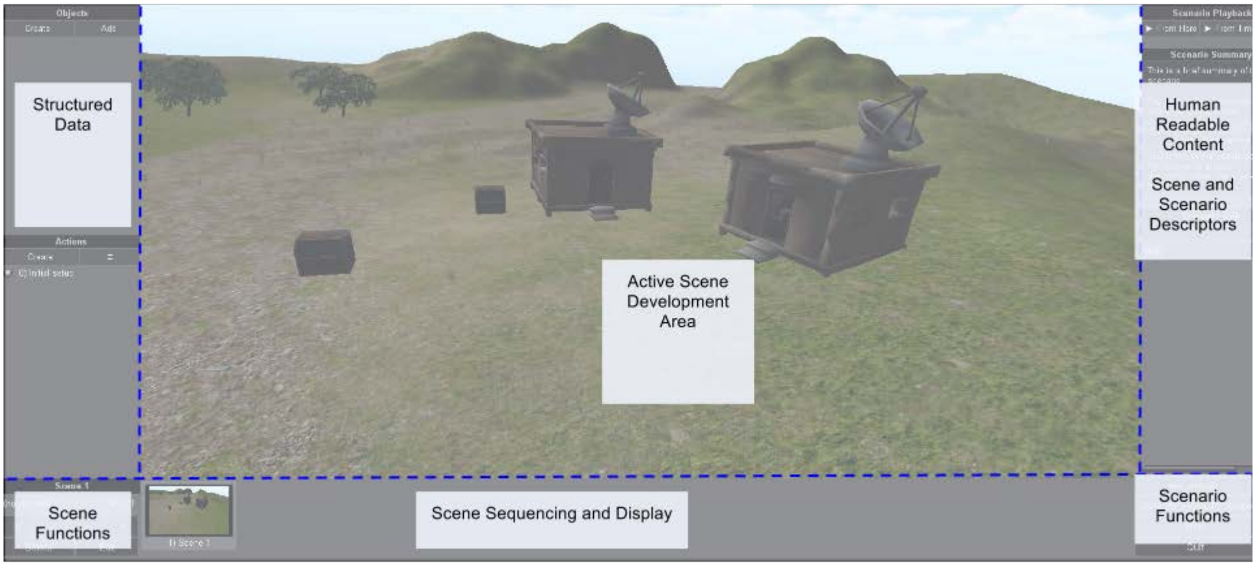

Figure 27 CONOPS Authoring Interface – User Elements

Within the scenario, the user can create, specify, add, and modify objects specific to the contact-to-fire scenario. Once the storyboard is complete, the user can do a playback from time zero and watch as the simulation unfolds. The current release of the prototype internalizes the health and ammunition measures, using a random number generator for health (if a soldier is hit with fire) and a uniform distribution for firing rates to determine ammunition remaining.

An example scenario building sequence and the environment representations are shown below. The scenario is simplified to highlight the functionality of the software and its flexibility.

Initial Scene (Figure 28): The application provides the user with an uneven terrain, with some reinforced buildings shown, along with outbuildings and some flora. The buildings are 3D models, and the author can use the keyboard and mouse to circle and examine them.

Figure 28 Initial Scene = Battlefield domain

Once the scene is opened, the author can add personnel, both friendly and enemy soldiers. The menu for object addition also contains some placeholders as well as the abstract class of “Team.” (Figure 29) The placeholders serve as exactly that – objects for which there is no 3D representation available in the domain yet, but for which the author needs a presence in the scenario. The resulting SysML will show the placeholder – and this can serve as an alert to the domain programmer that representation is needed. The abstract class of team can have a fluid definition. A team can be a squad, a battalion or, in this case, two soldiers. What is powerful about this representation is that actions can be assigned to the team and all members will perform them – crouch, crawl, walk, run, stand, etc.

When adding personnel, they are instantiated with 100% health and 100% ammunition ratings (Figure 29). As the soldiers fire and/or are shot, their health may or may not be reduced and if they are shooting, their ammunition remaining will be reduced.

Figure 29 Creation of characters

The author can easily change the camera angles, as well as pan, zoom in, zoom out, rotate and view all of the surrounding area. For this scenario, there are two enemy soldiers and two friendly soldiers in this scene. Figure 30 shows the Objects currently residing in the scene. You can see that there is also a Team associated with the scene – there is no physical representation but the object exists.

Figure 30 Character listing for scene

Figure 31 Scene with Soldiers in place

In Figure 31 the Agri-enemy soldier has a red arrow over his head – the red arrow indicates that the character is “selected.” The Agri-enemy is selected using a radio button (Figure 32). This selection means that we can now create an action to be associated with that character.

Figure 32 Action Listing for Character

In this example, the action Moves to () for Agri-enemy is selected using a radio button (Figure 32). The verb "moves" is used differently when applied to different characters – there is a "Moves" for an Item. In this case, "Moves to" will assume that the author selects a locale and moves the soldier there. One subtlety which arises when assigning actions is that although the basic "sentence" structure of an activity may be the same, the originators and recipients of that action may be more than simple person-to-person action – it may be the actions of a team toward one person, one person toward a team, or a team toward another team. In order that the display correctly depicts a multitude of possibilities, this construction needed to be considered in the application development.

Figure 33 Action with indicated object

The Actor is selected (Figure 33), then moved to the next position. When the simulation is run, the character will stop at this point (Figure 34).

Figure 34 Move Action for Agri-enemy completed

Similarly, the operator can indicate that Bob will shoot Agri-enemy, and that Steve will move behind the far bunker (Figure 35).

Figure 35 Repositioning of soldier

Finally, we have Bob move to the far outbuildings, visible in the upper right hand side of Figure 36.

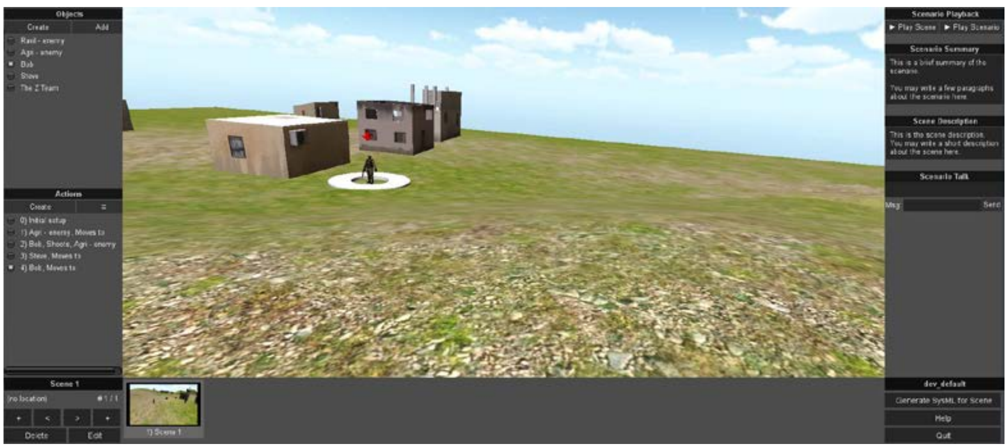

Figure 36 Character moved to far edge of terrain

In addition to the listing of actions which constitute the scenes unfolding, the application is capable of adding scenes which will then continue the action. In this case, if the action has now moved to the outbuildings and will revolve around the character Bob, the author can now add a new scene by pressing the + in the Scene Function area, and a new thumbnail will appear on the bottom Scene Sequencing and Display area (Figure 37).

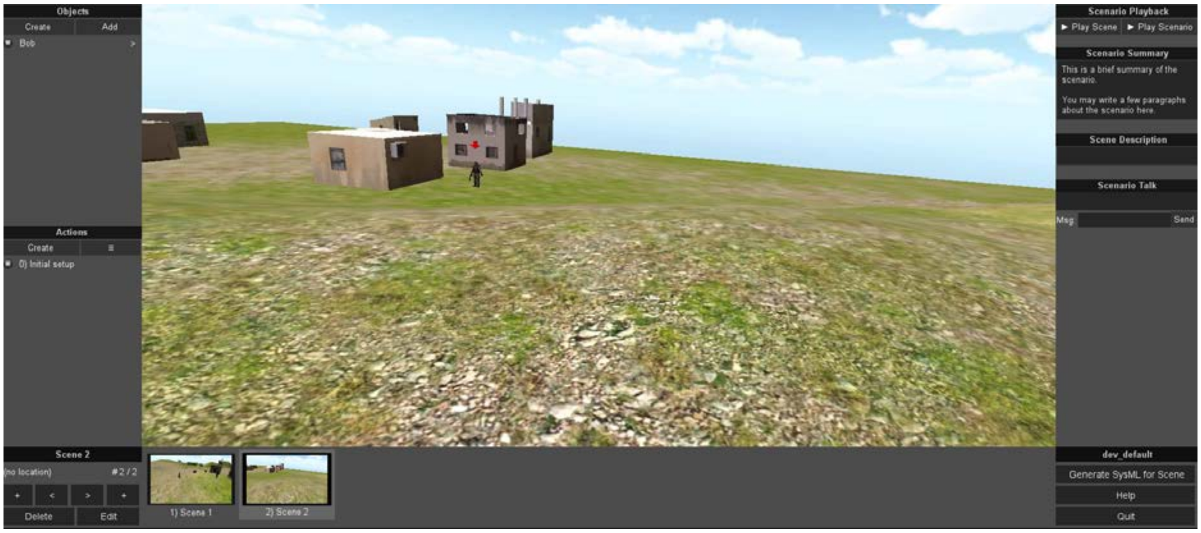

Figure 37 Scene 2 added at the second locale

The format of this report does not allow for the demonstration of animation, or to show the relatively fluid flow from one scene to the next.

3.4.6 Database Selection and Other Dependencies

Unity is based on the Mono Project[1], an open source implementation of the .Net framework. Therefore, it is possible to use common .Net libraries directly in Unity. Mono allows developers to create cross-platform applications easily, using C# and the Common Language Runtime (CLR) that is binary compatible with .Net. Mono runs on Linux, MS-Windows, Mac OS X, BSD, Sun Solaris, Nintendo Wii, Sony PlayStation 3, and Apple iPhone. It will also run on x86, x86-64, IA64, PowerPC, SPARC(32), ARM, Alpha, s390, s390x (32 and 64 bits). The use of Mono and the CLR means that any language that can compile to pure Intermediate Language (IL, sometimes referenced as CIL) should work under Mono. Some Mono-compatible compilers include C#, Python, Java, Scala, Boo, Visual Basic .Net, and Cobra. IL itself is in a binary format and is not readable by humans.

Another software dependency of the ICEF is Apache CouchDB[2], an open source database project developed in the Erlang programming language with a web-based (http) API. It is available for most platforms. Apache CouchDB was chosen for several reasons:

- Erlang is used to build massively scalable real-time systems which require high availability and reliability. Erlang is extensively used in banking, telecom, e-commerce, computer telephone and instant messaging applications.

[1] http://mono-project.com/
[2] http://couchdb.apache.org/

- Documents are the primary unit of data in Couch DB, and can consist of any number of fields and attachments – this enables the developer to associate 3D models with extensive object information in an encapsulated, efficient fashion.

- The CouchDB document update model is lockless and optimistic. Authors using client applications save changes to the database - if another client editing the same document at the same time saves their changes first, the author gets and edit conflict error when saving. This gives the author the change to evaluate the other client's changes, and either accept them or update the document and re-try the update.

- The database is always in a consistent state; CouchDB will never overwrite committed data or associated structures.

- CouchDB enables efficient building of views, because the data is stored in semi-structured documents, rather than spread across numerous records and tables. This is of particular interest to this research task.

4 RESEARCH/QUESTIONS AND LESSONS LEARNED

Like RT 30, RT31 evolved from the initial CONOPS research task, RT3. Due to the relatedness between each of these research tasks the research team defined a high level research question to tie together the RT 3/30/31 thread. The derived research question was:

Can the process of Concept Engineering improve the understanding and development of a concept of operations using gaming technologies along with an interactive, collaborative, and graphical environment?

From this question, each research task contains lower level research questions to address specific task goals.

4.1 RESEARCH QUESTIONS FOR RT31A

- Can the process of CONOPS modeling and simulation be improved through the use of a graphical user interface which would serve as a conduit for data?
- What are the benefits of a single user interface for the tools currently in use for modeling and simulation studies?
- What are the drawbacks of a single user interface for the tools currently in use for modeling and simulation studies?
- Does real-time collaboration between distributed stakeholders improve the CONOPS development in the area of modeling and simulation?
- Can a real-time collaboration environment enable quicker consensus on CONOPS generation?
- Are there new or specific issues in asynchronous software development in an immersive environment?

4.2 RESEARCH LESSONS LEARNED

As before, any advanced investigation must be supported by the software – and although the software effort is secondary to the research questions being posed, it continues to consume a great deal of the effort. Although most of the previous "lessons learned" from RT-31 still hold, several new lessons were learned during this follow-on effort:

4.2.1 Project Management Lessons Learned during RT31a

- The Presagis battle simulation package is highly complex and requires a great deal of time and training to generate output capable of being used by the current application. Extensive time and effort was spent in configuring a computer laptop for the research team – it was hoped that it would be capable of running Presagis and accepting input from the ICEF application. This effort required so much time that pursuing this line of research was taking resources away from the main research thrust. It was decided to concentrate our investigations into increasing the fidelity of the use-case scenario building.

- The use of Trello, an online project management tool, was invaluable in managing task assignments between the two programming locations. By attaching tasks to various build cycles, each team was able to stay on target with their deliverables.

- Weekly meetings were helpful not only in tasking but when discussing difficulties encountered by either team. It is strongly suggested that multi-site development always adopt a weekly review of both code and project plan.

4.2.2 Architecture Lessons Learned during RT31a

- One major addition during this phase of the project was the integration of the RT-31a scenario into the existing architectural framework used by RT-30a. Several items arose during this integration:
    - Performance was not a strong focus here, proof-of-concept was the driving consideration.
    - In the process of integration, it was inevitable that some refactoring of code would occur. This required additional test time to validate the robustness of the code.
    - The domain of RT-30a, which of necessity had several differing locales required the use of multiple cameras. RT-31a, in contrast, is a single locale viewed across several time periods. This alternate view of a scenario required additional modifications to the data structure as well as the overall architecture.

4.2.3 Project Code/Construction Lessons Learned during RT31a

- Asset Server Change Control/Staging Platform
    - Individual project access in asset server still needs to be transparent to all developers.
    - There were frequent slow-downs when committing to or updating from the Asset Server, this was especially noticed by the Auburn team.

- Highly-modular design still vies with programming strategies – optimal breakpoints are difficult to identify
    - Assignment of modular design elements can be problematic and, because of the iterative nature of design and development, is a real challenge
    - The graphic design of 3D models, including scaling and manipulation, took longer than originally anticipated; this is not due to the provenance of the models, but is inherent to 3D environments
    - Adequate 3D models may not be commercially available, which can hamper development efforts while one is built.

- As in all the Unity-based software, programmers are encouraged to avoid manipulation of the object surface meshes – in order to indicate a "selection" insert an indicator above the object itself

- Actual physical movement of groups (for instance, a group of ground vehicles, personnel, or deployment of fleet) is quite intricate. Additional time is needed to coordinate all the movements and storage of that data.

- Containment of objects can impact performance, storage, and retrieval. Due to the small number of objects manipulated, this wasn't seen in these applications, but it may become a larger issue for more populated scenarios.

- Drag and drop functionality was implemented to allow for easy placement and movement of objects in the scene. The user was given the ability to move the camera around at their discretion using the "W, S, A, and D" keys. The viewing angle could be more easily controlled by the mouse. This gave the user a great deal more ability to navigate the CONOPS environment.

- Project coding standards, naming, and organization became more critical, especially since the development was shared between two geographically-diverse coding teams, with differing levels of Unity experience. This was seen in both parallel development and with the code management provided by the Asset Server.

- Although RT30a and RT31a now share a common framework and architecture, the domain and thrust of analysis was different for each research task. Although the sample space of experience is small (2 domains), it is clear that adding a second, somewhat similar domain type does not noticeably lengthen development time. If the environment being modeled is radically different physically (underwater, atmospheric) this may not be the case, but for these environments, it was not excessively time-consuming.

- The time-consuming work resulted from the somewhat specific considerations of the domain – being one of a combat contact-to-fire scenario rather than a collection of street scenes. Verb lists, interactions, the fluid flow of scenes, and the specifics of military collaboration/conflict (time-dependent and activity-dependent health and supply monitoring) required additional design and display considerations.

- When looking at domain-initiation considerations, it was necessary to examine the loading of 3D models dynamically during run-time. It was determined that Unity does support the importation of objects at run-time via the use of Asset Bundles. These are a Unity file type that consists of grouping of like files belonging to a single object (such as a 3D model of a soldier complete with animation, wireframe, color palette, meshes, etc.).

- There is a distinct lack of free high-quality 3D models available, that are also capable of sophisticated motion. This is not true of static "window-dressing" models, or wall treatments. Care must be taken if the sponsor plans to distribute models, whether free or purchased. This may also impact further development if this effort is transitioned to an open-source environment.

5 CONCLUSIONS

The goal of this research task was to continue the research and effort required to develop a concept engineering software demonstrator that enabled soldiers to develop a limited set of scenarios centered on squad operations. The work was to extend the Integrated Concept Engineering Framework (ICEF) prototype developed for RT30/31.

At the conclusion of this research task, RT30a and RT31a are based on a common architecture framework. This framework now allows for the addition of new domains to facilitate the development of a CONOPS in any number of domains. While the effort was based on a generic landscape and soldier domain, the architecture now allows a development team to create and utilize other domains – such as an urban warfare domain, a jungle warfare domain, or even a domain that represents a military installation.

This research demonstrated that it is possible to utilize the strength of a 3D game development environment to create a graphical CONOPS creation tool that is easy for a soldier to use. Appendix B, while not part of this specific research task, demonstrated that the use of this type of tool improved the shared mental model, and the quality of the developed CONOPS by individuals.

Finally, the research team was able to demonstrate that the output of the CONOPS, in the form of actors, objects, and activities can be exported to an XML file. This file can then be imported into a SysML tool. This is significant in that the CONOPS can now be the basis of the operational architecture.

APPENDICES

APPENDIX A: REFERENCES

Peters, R. D. (n.d.). *Ideal Lift Kinematics - Complete Equations for Plotting Optimum Motion*. Retrieved January 2012, from www.peters-research.com: http://www.peters-research.com/index.php?option=com_content&view=article&id=61%3Aideal-lift-kinematics&catid=3%3Apapers&Itemid=1

Esafahbod, Behnam, Integrated Concept Engineering Framework – a Visually-Rich Stakeholder-Centric System Design Software, Master's Thesis, Stevens Institute of Technology, May 2013.

APPENDIX B: RT 30 RESEARCH SURVEY AND ANALYSIS

To begin to study the effectiveness of the Integrated Concept Engineering Framework (ICEF), an extensive literature review was conducted to discover metrics that exist for evaluating concept engineering tools and processes. While a fully formed metrics and evaluation scheme that fit the needs of this research had not been previously created, there was considerable investigation of assessment techniques for collaborative problem solving, as well as indicators of shortcomings of CONOPS that need to be addressed. Given this research, a set of metrics was developed to assess concept engineering. These metrics were separated into:

- artifact metrics – to enable CONOS-specific assessment
- collaboration metrics – to study how users and engineers work together to develop a CONOPS
- experience metrics – to measure the effect that different professional and life experiences have on CONOPS development

The metrics are summarized below (Table 5, Table 6, and Table 7) and are further described in (Korfiatis, 2013). Each metric was used to derive a survey question to be delivered during CONOPS experiments, discussed below.

Table 5 Artifact metrics

Metric/Source	**Definition**	**Survey**
Understandable	How easy it is to understand artifact	• The final artifacts provide a sense of the overall scenario • The final artifacts are easy to understand
Balanced Point of View (IEEE, 1998)	How well the artifacts represent a collection of individual PoVs How well do users express expectations through the artifact	• The final artifacts represent an acceptable balance between all of the group's needs • The final artifacts favor the needs of one stakeholder over another
Accuracy	How accurately artifacts represent scenarios. CONOPS must provide accurate descriptions of needs	• The final artifacts clearly represent needs of your role • The final artifacts clearly represent an accurate portrayal of the group's negotiated scenarios
Applicability to System Design	How useful the artifact would be to future developers A textual document tends to be cumbersome and of little use as a communication tool between stakeholders and developers	• The final artifacts provide clear guidance to system designers for system development • The final artifacts provide a useful tool to promote future conversation between stakeholders • The final artifacts be useful for educating new stakeholders later in the development process
CONOPS Elements (Fletcher, 2012; Roberts, 2008; Saldana, 2012)	Does the artifact include CONOPS elements that are required in CONOPS standards but shown to be under-addressed in traditional CONOPS?	• The final artifacts represent human roles • The final artifacts clearly represent the number and type of personnel required for scenarios • The final artifacts clearly represent personnel interfaces in the scenarios
Maintainability and Evolve-ability	How easily the artifact could be maintained, updated or evolved CONOPS should be updated to reflect evolving situation Amending textual CONOPS can be time consuming and lead to inconsistency across document	• The final artifacts are easy to edit if stakeholder needs were to change • The final artifacts are easy to edit to address new stakeholder needs

Table 6 Collaboration metrics

Metric/Source	**Definition**	**Survey**
Reduce Development Time (Mostashari, McComb, Kennedy, Cloutier, & Korfiatis, 2011)	Time required to develop CONOPS	• The time required to produce the scenarios is reasonable given the quality of the final artifacts
Satisfaction with Collaboration	Indicates that stakeholders leave with a sense of satisfaction that the collaboration was effective	• You are satisfied that the final scenarios address your need. • You are satisfied with the collaborative exchange during scenario development
Mutual Understanding (D. F. Noble, 2004)	The extent to which team members agree or disagree	• Your needs were adequately understood by the group • You adequately understood the needs of other group members • During scenario development your groups was able to correct misconceptions on each other's need
Communication (Fletcher, 2012) (Linebarger, Scholand, Ehlen, & Procopio, 2005)	How was communication between team members affected by the use of a specific scenario development process	• The scenario development process led to clear and unambiguous conversation about the scenarios • The scenario development process promoted critical dialog and skepticism • Verbal communication was improved through using the scenario artifacts
Shared Mental Model (Cloutier et al., 2010; McComb, 2007)	A common internal representation of the world, an event or a user scenario that is shared between team members	• A shared vision of the problem was reached by the group during the development process • A shared vision of the solution was reached by the group during the scenario development process • A shared vision of typical user scenarios was reached by the group during the scenario development process

Group Problem Solving (D. Noble & Kirzl, 2003)	Group problem solving is a major subset of team activities presented in the Framework for Collaboration Model (D. Noble & Kirzl, 2003). Five major types of group interaction can take place during group problem solving	**Brainstorming** • Adequate consideration was given to alternatives during scenario development • By developing the scenario artifacts, alternatives were discussed that would not have been otherwise discovered through conversation • A broad range of solutions were considered by the group that were relevant to scenario development **Prioritization** • The scenario development approach helped the group explicitly prioritize stakeholder needs • Any prioritization that took place during scenario development is reflected in the final scenario artifacts • The scenario development approach helped the group implicitly prioritize stakeholder needs **Discovering differences** • During scenario development, fundamental differences in stakeholder needs were discovered and discussed • By developing the scenario artifacts, differences that were discovered that would not have been otherwise discovered through simple conversation **Negotiation** • By developing the scenario artifacts, there was more opportunity for meaningful negotiation than there would have been through simple conversation • Each stakeholder was able to fully present and explain their needs during scenario development. • One stakeholder's needs dominated the scenario development process **Consensus** • The scenario development process has allowed our group to reach a consensus on scenarios that everyone agrees with. • By developing the scenario artifacts, there was a greater level of consensus in the final scenarios than there would have been through simple conversation
Collaborative Macro-Cognitive Process (Letsky, Warner, Fiore, Rosen, & Salas, 2007; Warner, Letsky, & Cowen, 2005)	"the internalized and externalized high-level mental processes employed by teams to create new knowledge during collaborative problem solving" (Letsky et al., 2007)	**Adapted from Warner et al. (2005) to be captured and codified by observers:** • Visualization & Representation - presenting information in pre-processed forms • Building Common Ground - sharing common or joint knowledge and beliefs to build common ground (• Knowledge Sharing and Transfer - information is given by one person and received by another • Team Shared Understanding - synthesis of essential knowledge, held collectively by some and/or all team members • Solution option Generation - generating set of decision alternatives that satisfy the requirements of the task • Negotiation of Solution Alternatives - discussion to construct something new which neither individual could create on their own • Team Pattern Recognition - process of recognizing patterns in information, solution options or problem space • Converge individual mental model to team mm - convincing others to accept your data, information or knowledge • Critical Thinking - reflective reasoning about beliefs and actions • Mental Simulation - using mental models to make inferences about future states of a situation (what if...) • Intuitive decision making - A team rapidly reaching intuitive consensus • Compare solution against goals - discuss a final solution option against the goal • Analyze and Revise solution Options - analyze final solution options and revise them if necessary

Table 7 Experience metrics

Metric	Definition	Survey
Gaming Experience	Level of comfort in playing or creating video games or gaming engines, 3D immersive environments or other advanced visualizations	Rate your comfort with the following concepts/activities: • Game playing • Game development • Visualization
Systems Engineering Experience	Work experience and level of comfort in systems engineering related activities	• How many years of experience do you have in systems engineering? • How many systems engineering projects would you estimate you have been involved in? Rate your comfort with the following concepts/activities: • Systems Engineering • Model Based Systems Engineering • System Design • Modeling and Simulation
CONOPS Experience	Exposure, work experience and level of comfort to CONOP documents and CONOPS/Concept Development	• How many CONOPS development processes have you participated in? • How many CONOPS documents have you read? Rate your comfort with the following concepts/activities: • Concept Development • CONOPS Development • Requirements Elicitation/Management

ICEF Experimental Procedure

Two laboratory experiments were conducted to study the effectiveness of ICEF. Both experiments involved participants producing artifacts representing the operational scenario section of the CONOPS document. All groups were presented with a number of written descriptions of a news agency scenario and asked to either model operational scenarios using the ICEF tool or create a text based narrative akin to the traditional CONOPS development process. Due to limitations placed on the experimental design by theRT30 research sponsor, there was no control group for the first experiment. Attendance in this first experiment consisted of twenty-one DoD systems and software engineers, development and operations personnel, technical writers, and managers separated into five groups. The experiment was conducted as displayed in the SysML activity diagram in Figure 38.

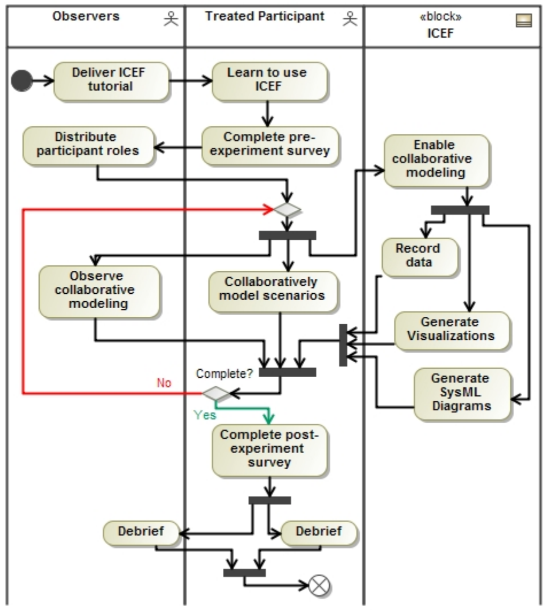

Figure 38 ICEF experiment 1 procedure

A brief instructional tutorial on the functionality of the ICEF tool was presented to all participants by the project manager and a primary ICEF software developer. After fifteen minutes of instruction, all participants felt comfortable with the functionality of ICEF and a pre-experiment survey was administered to record the participants' level of experience in CONOPS development, gaming, visualization and systems engineering. Following the completion of the pre-experiment survey, each participant received a participant instruction handout, which provided general instructions on the experiment, background information on typical operation of a news agency, and a description of the five specific scenarios to be modeled. A list of roles (news editor, reporter, systems engineer, support asset manager, and acquisitions and support personnel) was provided in the participant instruction handout. The groups were allowed to assign roles using any method in which they decided and each participant was provided with a role specific handout. These handouts were written to inform each participant with the needs, concerns and responsibilities of their roles, which were purposely set to be contradictory.

Once the role-specific handouts were distributed, the experiment began. Each group was responsible for collaboratively modeling as many of the scenarios as they could manage using ICEF. Their primary objective was to be sure that their roles' needs and concerns were evident within the model. Because the first experiment took place using DoD employees, audio recording of the modeling sessions was not possible. Therefore, during the first experiment, five observers used a structured observer rubric to evaluate the type of collaborative cognitive processes each group was undergoing. At the end of the session, each team produced animated visualizations and SysML diagrams featuring the operational scenarios as modeled using the ICEF. Finally, participants were asked to complete a post-experiment.

The second experiment was run in a similar fashion using twenty-five Engineering Management undergraduate students from a third-year Engineering Design class. While not active in the systems engineering domain professionally, these students had recently concluded coursework related to CONOPS development and requirements elicitation taught by a systems engineering professor with numerous years of practical industry experience. The students were separated into eight groups. After being divided, four of the groups were asked to move to a separate room where they were subjected to the same experimental procedure utilized in the first experiment. The remaining four groups acted as control groups and did not receive information about the ICEF. Instead, they were asked to develop a textual description of the same five news agency scenarios. The result of their discussions was a Microsoft Word document containing a narrative describing the operational scenarios, comparable to the current CONOPS artifact and development process. The use of student allowed for audio recording of the sessions, and as such, the research team recorded conversations by each of the groups. After the experiment, the same five observers were asked to listen to the recordings and codify the types of macro-cognitive collaborative processes taking place during the groups CONOPS development session. A detailed look at the second experimental procedure is seen in Figure 39.

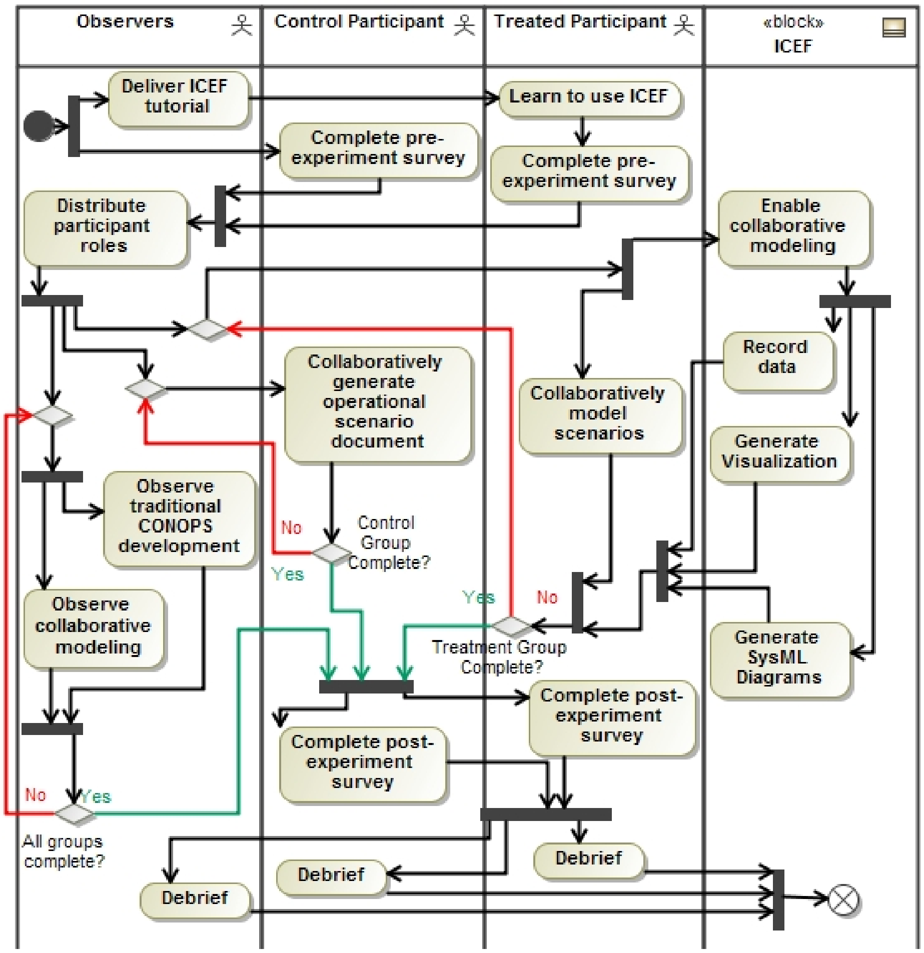

Figure 39 Experiment 2 activity diagram

Data Collection

As briefly described above, data was collected during the experiment using three methods (Figure 40).

- Surveys were used to elicit feedback directly from experiment participants. To establish a baseline on the background, expertise and comfort level of participants in the fields of systems engineering, CONOPS development and gaming, visualization and immersive environments, the pre-experiment survey asked participants to select a pre-determined range of values describing their years of systems engineering experience, the number of systems engineering projects they have worked on and the number of CONOPS they have read and developed. Participants also ranked their experience in topics related to this research as Very Experienced, Some Experience or No Experience. A post-experiment survey was also distributed to evaluate ICEF's effectiveness. The survey was designed to assess the participant's perception of the collaborative modeling process and the resultant CONOPS artifacts.

- The method of observation used in this experiment was grounded in established models for measuring cognitive processes during collaboration. Based on previous collaboration research, observation was centered on classifying the types of macro-cognitive processes used by participants during the collaborative scenario modeling. The collaboration model used attempts to measure how many instances of specific cognitive processes occur, how often they are encountered and when they transpire. To reduce possible observer bias, each group of participants was observed by two observers at a time and the observers rotated groups every twenty minutes. Differences in scoring were discussed and reconciled between the observers. Additionally, the database and logging function of each ICEF system provided researchers with the ability to recreate and document what occurred in the software while specific macro-cognitive processes were taking place. Observers were also responsible for collecting qualitative observations of individual and group behaviors during collaboration.

- The ICEF was specifically designed to capture information regarding how the users interacted with the software. This was carried out using internal activity logging. The activity log serves a number of purposes including measuring timing between modeling activities and recording the placement and deletion of objects, relationships and attributes.

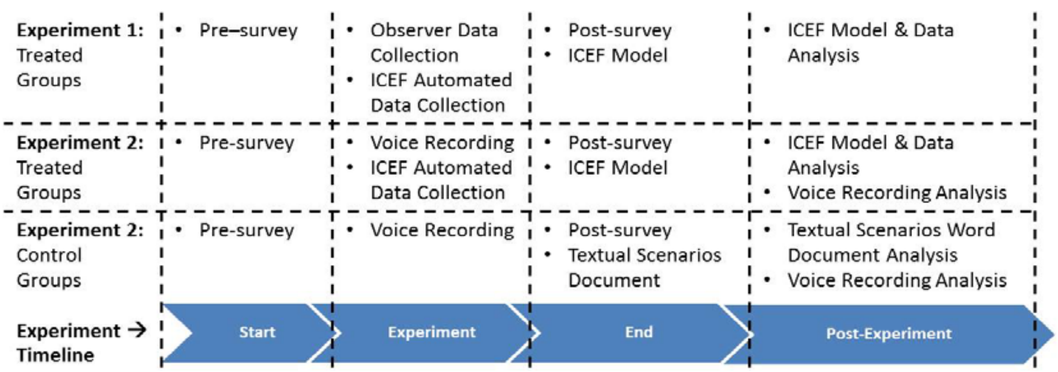

Figure 40 Experiment data collection

RESEARCH HYPOTHESES

CONOPS can be examined in terms of both collaboration during development, and the final artifact. To this end, two hypotheses were developed to drive data collection and analysis.

- Hypothesis 1: Use of the ICEF will improve the operational scenarios artifact of a CONOPS.
- Hypothesis 2: Use of the ICEF will improve collaboration during the development of the operational scenarios section of a CONOPS.

BAYESIAN DATA ANALYSIS

Given the scope of this research and the design of experiments described above, there were limiting factors in selecting an appropriate data analysis technique. These include the fact that:

- few recognized metrics have been established or data collected and published concerning CONOPS development and concept engineering
- data collection lead to both qualitative and quantitative data so the analysis technique should be able to handle both types of data
- the sample size of the experiment was relatively small and was not fixed across experiments

Given these limitations, Bayesian Hypothesis Testing was selected for data analysis. In-depth discussion of Bayes' theorem and Bayesian data analysis is beyond the scope of this report. A full treatment of Bayesian data analysis can be found in (Fenton & Neil, 2012; J. Kruschke, 2010).

Bayesian analysis was selected here because it is fairly accurate with smaller sample sizes (Uusitalo, 2007), it does not require a fixed sample size across experiments (J.K. Kruschke, 2010) and it is effective in combining both experimental and observation data (Cooper & Yoo, 1999). Additionally, since there is little previous published work on concept engineering, the Bayesian approach is well positioned to capture this uncertainty and treat it explicitly (Uusitalo, 2007).

As described in (Fenton & Neil, 2012; J. Kruschke, 2010), Bayesian hypothesis testing is easily conducted using a Bayesian network. A Bayesian network is a directed acyclic graphical representation of a set of variables and their relationship to each other. The network nodes can represent variables, parameters, hypotheses or observed data, and the directed edges describe the relationships between these variables. In Figure 41, the metrics described above were added as middle level nodes in the Bayesian network (green nodes). Specific data from surveys and other sources were added as input nodes (yellow nodes). Two nodes were created to represent each hypothesis, and connected to a final output node labeled Combined Output (orange rectangular nodes).

For this research, due to lack of established prior data, it was assumed that the experiments described above would result in one of three distinct outcomes, each of which can be seen as a competing hypothesis:

- Alternative Hypothesis 1 (H_{A1}) - The observed data shows evidence that the ICEF was ineffective in improving CONOPS artifacts and collaboration
- Alternative Hypothesis (H_{A2}) - The observed data cannot be used to make a judgment as to the effectiveness of the ICEF in improving CONOPS artifacts and collaboration
- Alternative Hypothesis 3 (H_{A3}) - The observed data shows evidence that ICEF was effective in improving CONOPS artifacts and collaboration

Based on (J. Kruschke, 2010), if the data gathered from a group using the ICEF is inputted to the Bayesian network and the resulting probability distribution:

- fits entirely within the H_{A1} distribution, the data has a high probability of supporting the conclusion that ICEF was ineffective
- fits entirely within the H_{A3} distribution, the data has a high probability of supporting the conclusion that ICEF was effective

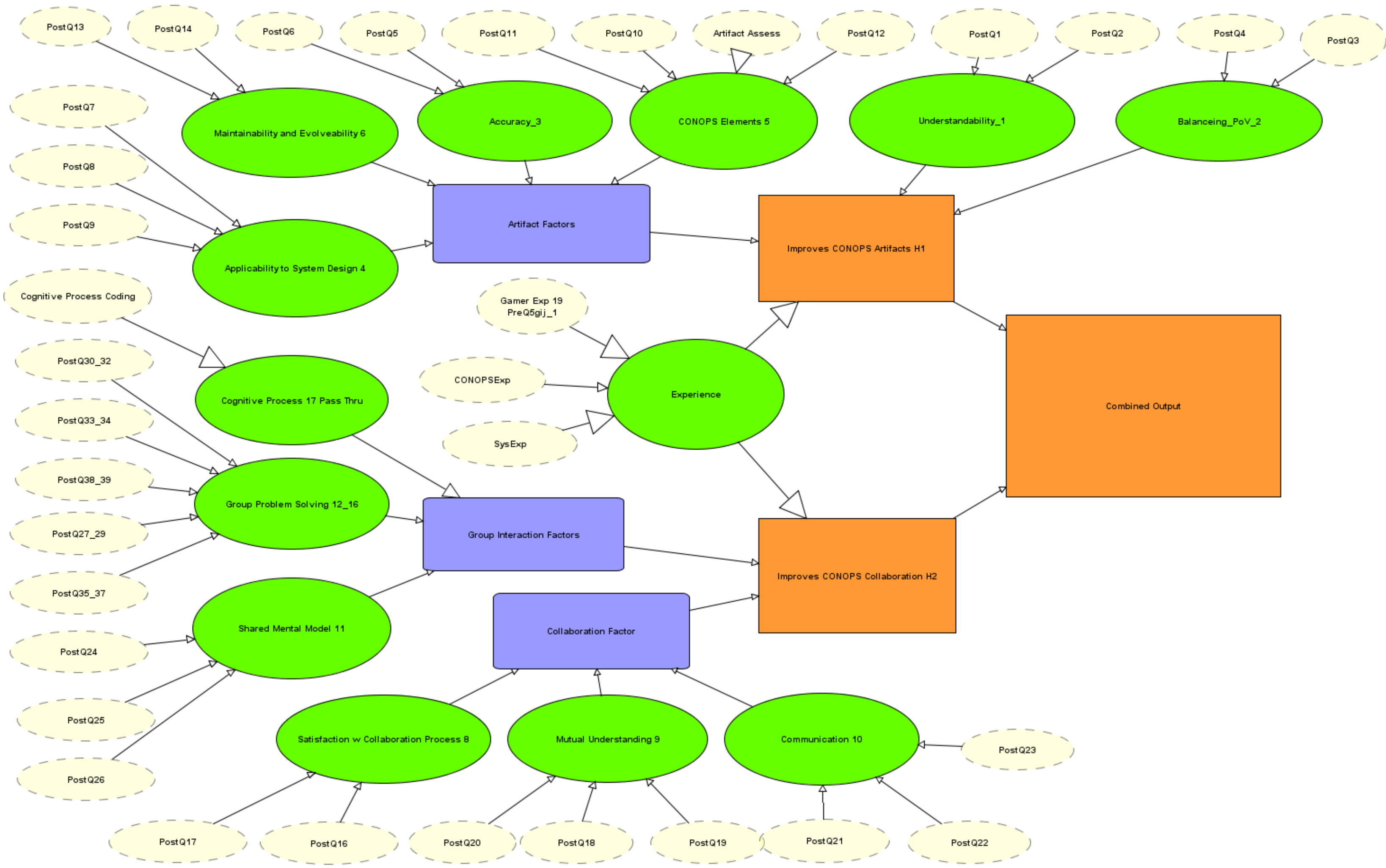

Figure 41 ICEF Bayesian network

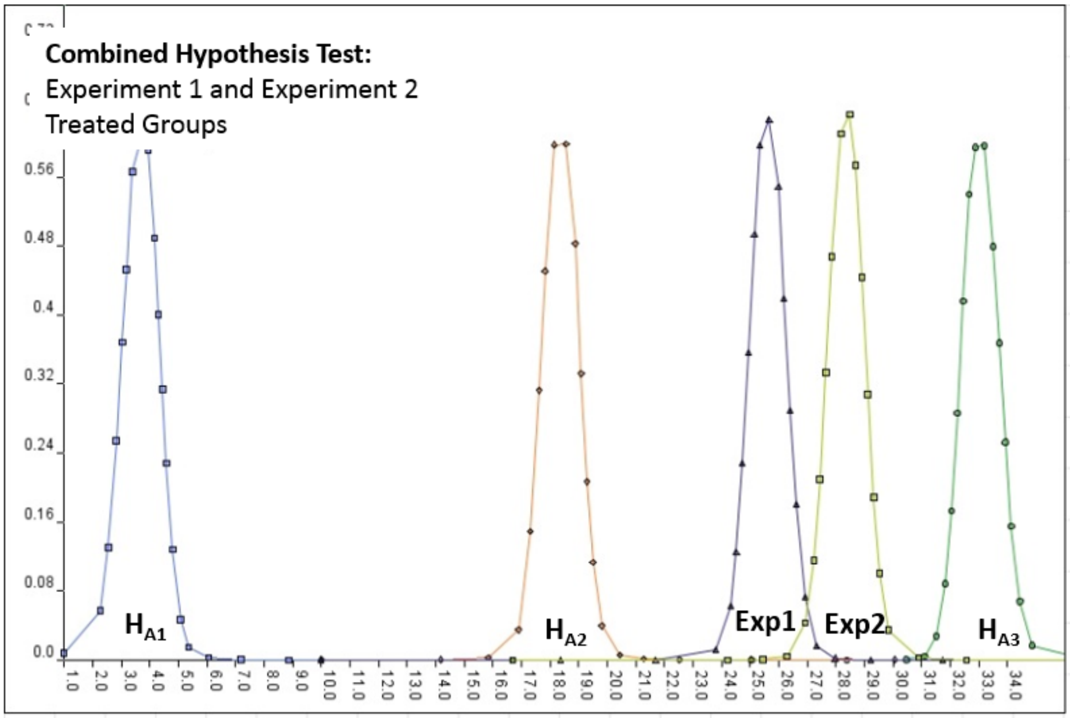

Figure 42 Bayesian hypothesis test

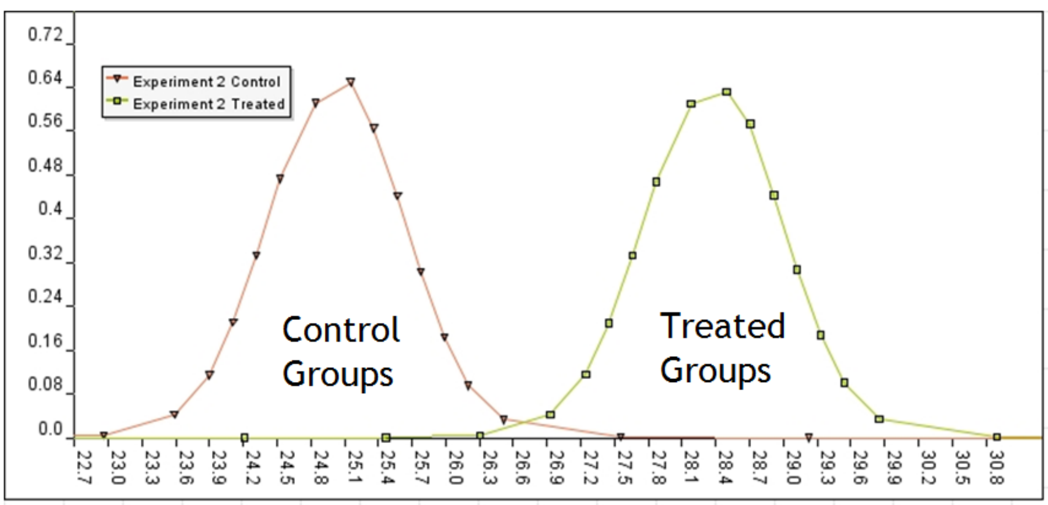

Figure 43 Experiment 2 comparative analysis

EXPERIMENT RESULTS

Figure 42 shows the results of the Bayesian analysis using data collected from both treated groups in the first and second experiment. Each set of observed data formed a probability distribution that lies between H_{A2} and H_{A3}. Because the distribution does not fit well within any hypothesis distribution, none of the alternate hypotheses developed can be accepted with full confidence. However, Bayesian hypothesis testing demonstrates coherence (Wagenmakers et al., 2010), meaning that the position of each distribution can be directly compared to each other, and conclusions can be drawn based on the relative position, size or shape of a distribution. From Figure 42 we can see that the distributions of the observed data fall very far outside the distribution of H_{A1}. It can be stated with confidence that based on the data collected from these experiment there is little to no evidence in favor of accepting H_{A1}, and it is far more likely that evidence exists to support H_{A3}. While this is not an outright acceptance of any alternative hypothesis put forth, the data collected in these experiments are much more likely to support the effectiveness of ICEF rather than its ineffectiveness. The proximity of the observed data's distribution to the H_{A3} posterior is an indicator of the level of confidence of this conclusion. At the same time, Figure 43 displays a comparison between the treated and control groups of the second experiment. The distribution of those who received the treatment (utilized the ICEF) and those who acted as the control group (did not utilize the ICEF) can be compared to one another directly. Based on this comparative analysis, the data from this experiment shows a preference for the ICEF approach over the traditional concept engineering approach.

APPENDIX B REFERENCES

Cloutier, R., Mostashari, A., McComb, S., Deshmukh, A., Wade, J., Kennedy, D., . . . Carrigy, A. (2010). Investigation of a Graphical CONOPS Development Environment for Agile Systems Engineering: Systems Engineering Research Center.

Cooper, G. F., & Yoo, C. (1999). *Causal discovery from a mixture of experimental and observational data*. Paper presented at the Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence, Stockholm, Sweden.

Fenton, N., & Neil, M. (2012). *Risk Assessment and Decision Analysis With Bayesian Networks*: Taylor & Francis.

Fletcher, D. (2012). *A Comparative Analysis on Development Methods of Traditional and Graphical CONOPS*. Masters, Stevens Institute of Technology, Hoboken, NJ.

IEEE. (1998). IEEE Guide for Information Technology , System Definition , Concept of Operations (ConOps) Document. In I. C. S. Software Engineering Standards Committee of the (Ed.), (Vol. 1362-1998 (R2007)).

Kendall, T., & Hutchins, S. G. (2009). *Use of a team collaboration model to identify candidate knowledge management and collaboration technologies*. Paper presented at the Proceedings of the 9th Bi-annual international conference on Naturalistic Decision Making.

Korfiatis, P. (2013). *Development of a Virtual Concept Engineering Process to Extend Model-Based Systems Engineering.* PhD Dissertation, Stevens Institute of Technology, Hoboken, NJ.

Kruschke, J. (2010). *Doing Bayesian Data Analysis: A Tutorial Introduction with R*: Elsevier Science.

Kruschke, J. K. (2010). Bayesian data analysis. *Wiley Interdisciplinary Reviews: Cognitive Science, 1*(5), 658-676. doi: 10.1002/wcs.72

Kruschke, J. K. (2010). What to believe: Bayesian methods for data analysis. *Trends in cognitive sciences, 14*(7), 293-300.

Letsky, M., Warner, N., Fiore, S. M., Rosen, M., & Salas, E. (2007, June 19-21). *Macrocognition in complex team problem solving.* Paper presented at the International Command and Control Research and Technology Symposium, Newport, RI.

Linebarger, J., Scholand, A., Ehlen, M., & Procopio, M. (2005). *Benefits of synchronous collaboration support for an application-centered analysis team working on complex problems: a case study.* Paper presented at the GROUP '05: Proceedings of the 2005 international ACM SIGGROUP conference on Supporting group work, Sanibel Island, Florida, USA.

Marek, J. D., & van der Gaag, L. C. (2000). Building Probabilistic Networks: 'Where Do the Numbers Come From?' Guest Editors' Introduction. *IEEE Transactions on Knowledge and Data Engineering, 12*(4), 481-486.

Masson, M. J. (2011). A tutorial on a practical Bayesian alternative to null-hypothesis significance testing. *Behavior Research Methods, 43*(3), 679-690. doi: 10.3758/s13428-010-0049-5

McComb, S. A. (2007). Mental model convergence: The shift from being an individual to being a team member. *Research in Multi Level Issues, 6*, 95-147.

Mostashari, A., McComb, S. A., Kennedy, D. M., Cloutier, R., & Korfiatis, P. (2011). Developing a Stakeholder-Assisted Agile CONOPS Development Process. *Systems Engineering, 15*(1), 1-13. doi: 10.1002/sys.20190

Noble, D., & Kirzl, J. (2003). *Objective Metrics for Evaluation of Collaborating Teams.* Paper presented at the Command and Control Research Technology Symposium, Washington, D.C.

Noble, D. F. (2004). Understanding and Applying the Cognitive Foundations of Effective Teamwork: Office of Naval Research.

Peters, R. D. Ideal Lift Kinematics - Complete Equations for Plotting Optimum Motion. *www.peters-research.com*. Retrieved from http://www.peters-research.com/index.php?option=com_content&view=article&id=61%3Aideal-lift-kinematics&catid=3%3Apapers&Itemid=1

Roberts, N., & Edson, R. (2008, October 21). *System Concept of Operations: Standards, Practices and Reality.* Paper presented at the NDIA 11th Annual Systems Engineering Conference, San Diego, CA.

Saldana, A. (2012). *Do Graphical CONOPS Help to Better Identify Integration and Testing Points, Planning, Estimation, and Overall Concept of Scenario?* Masters, Stevens Institute of Technology, Hoboken, NJ.

Uusitalo, L. (2007). Advantages and challenges of Bayesian networks in environmental modelling. *Ecological Modelling, 203*(3–4), 312-318. doi: http://dx.doi.org/10.1016/j.ecolmodel.2006.11.033

Wagenmakers, E.-J., Lodewyckx, T., Kuriyal, H., & Grasman, R. (2010). Bayesian hypothesis testing for psychologists: A tutorial on the Savage–Dickey method. *Cognitive Psychology, 60*(3), 158-189. doi: http://dx.doi.org/10.1016/j.cogpsych.2009.12.001

Warner, N., Letsky, M., & Cowen, M. (2005). Cognitive Model of Team Collaboration: Macro-Cognitive Focus. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting, 49*(3), 269-273. doi: 10.1177/154193120504900312